ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Τομέας Πυρηνικής Τεχνολογίας
Διευθυντής: Καθηγητής Σ.Ε. Σιμόπουλος

ΚΩΔΙΚΕΣ ΥΠΟΛΟΓΙΣΜΩΝ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB 7.8 (R2009a) ΚΑΙ SCILAB 5.2.2

COMPUTER CODES FOR THE CALCULATION OF THE THERMOPHYSICAL PROPERTIES OF LIGHT AND HEAVY WATER SUBSTANCES IN MATLAB 7.8 (R2009a) AND SCILAB 5.2.2 PROGRAMMING ENVIRONMENTS

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ ΤΟΥ
ΠΑΧΙΤΣΑ ΣΤΥΛΙΑΝΟΥ

Σπουδαστή της Σχολής
Μηχανολόγων Μηχανικών ΕΜΠ

Επίβλεψη: Λέκτορας Ν.Π. Πετρόπουλος

ΑΘΗΝΑ 2010

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΡΟΛΟΓΟΣ vi

ΠΕΡΙΛΗΨΗ viii

ABSTRACT.....x

ΚΕΦΑΛΑΙΟ 1: **ΕΙΣΑΓΩΓΗ**

1.1 **Γενικά**……..... 1-1

1.2 **Εξισώσεις για τις θερμοφυσικές ιδιότητες ελαφρού και βαρέος ύδατος**.. 1-2

1.3 **Δομή κωδίκων**…… 1-3

1.4 **Λειτουργία υποπρογραμμάτων**……. 1-3

1.5 **Προγραμματιστικά εργαλεία MATLAB και** SCILAB……..... 1-4

1.6 **Συμπεράσματα**……..... 1-4

1.7 **Παράρτημα**…… ... 1-5

ΚΕΦΑΛΑΙΟ 2: **ΣΥΝΤΟΜΗ ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΠΕΡΙΒΑΛΛΟΝΤΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB ΚΑΙ SCILAB**

2.1 **Εισαγωγή**…… 2-1

2.2 **Βασικά στοιχεία του MATLAB**……..... 2-2

2.3 **Βασικά στοιχεία του SCILAB**……..... 2-3

2.4 **"Τρέξιμο" κώδικα σε MATLAB ή SCILAB**…… 2-4

2.4.1 **Γενικά**..... 2-4

2.4.2 **"Τρέξιμο" MATLAB**..... 2-5

2.4.3 **"Τρέξιμο" SCILAB**..... 2-6

2.5 **Ανακεφαλαίωση**……..... 2-7

ΚΕΦΑΛΑΙΟ 3: **ΘΕΡΜΟΔΥΝΑΜΙΚΑ ΣΥΝΕΠΕΙΣ ΚΑΤΑΣΤΑΤΙΚΕΣ ΕΞΙΣΩΣΕΙΣ ΚΑΙ ΕΞΙΣΩΣΕΙΣ ΙΔΙΟΤΗΤΩΝ ΜΕΤΑΦΟΡΑΣ ΓΙΑ ΤΟ ΕΛΑΦΡΥ ΚΑΙ ΤΟ ΒΑΡΥ ΥΔΩΡ**

3.1 **Εισαγωγή**…… 3-1

3.2 **Θερμοδυναμικά συνεπείς καταστατικές εξισώσεις**……..... 3-1

3.3 **Εξισώσεις ιδιοτήτων μεταφοράς**…… 3-4

3.4 **Η περιοχή ισχύος των εξισώσεων**……..... 3-4

3.5 **Πίεση κορεσμού**…….. 3-6

3.6 **Υπολογισμός της πυκνότητας από τις καταστατικές εξισώσεις**……..... 3-7
3.7 **Υπολογισμός της θερμοκρασίας κορεσμού**……..... 3-13
3.8 **Χρήση των εξισώσεων σε κώδικα υπολογισμών**……..... 3-14
Διαγράμματα του 3ου Κεφαλαίου……..... 3-15

ΚΕΦΑΛΑΙΟ 4: **ΔΟΜΗ ΚΩΔΙΚΩΝ ΓΙΑ ΤΟΝ ΥΠΟΛΟΓΙΣΜΟ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΥΔΑΤΟΣ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ**

4.1 **Εισαγωγή**…… 4-1
4.2 **Γενικά χαρακτηριστικά κωδίκων**……..... 4-2
4.3 **Οδηγίες χρήσεως**..... 4-4
4.4 **Αντιμετώπιση λαθών (troubleshooting)**..... 4-7
4.5 **Υποπρογράμματα**…….. 4-8
4.5.1 **Υποπρογράμματα εισόδου – εξόδου**…… 4-8
4.5.1.1 **Υποπρόγραμμα (light_ / heavy_) wasp**……..... 4-9
4.5.1.2 **Υποπρόγραμμα checkpt**……..... 4-9
4.5.1.3**Υποπρόγραμμα fps**…….. 4-10
4.5.1.4 **Υποπρόγραμμα prl**……..... 4-10
4.5.2 **Υποπρογράμματα υπολογισμού θερμοκρασίας κορεσμού "**…… ... 4-10
4.5.2.1 **Υποπρόγραμμα fts**…….. 4-10
4.5.2.2 **Υποπρόγραμμα dfps**……..... 4-10
4.5.3 **Υποπρογράμματα υπολογισμού ειδικού όγκου**…… 4-10
4.5.3.1 **Υποπρόγραμμα densf**……..... 4-11
4.5.3.2 **Υποπρόγραμμα densg**…… .. 4-11
4.5.3.3 **Υποπρόγραμμα svlwl**…… ... 4-11
4.5.3.4 **Υποπρόγραμμα fpr4**…… 4-12
4.5.3.5 **Υποπρόγραμμα dfpr4**……... 4-12
4.5.3.6 **Υποπρόγραμμα svlwv**……... 4-12
4.5.3.7 **Υποπρόγραμμα fpr3**……..... 4-12
4.5.3.8 **Υποπρόγραμμα dfpr3**……..... 4-13
4.5.3.9 **Υποπρόγραμμα fp** ……..... 4-13
4.5.3.10 **Υποπρόγραμμα dfpd**…… .. 4-13

4.5.3.11 **Υποπρόγραμμα qmust**…… 4-13
4.5.4 **Υποπρογράμματα υπολογισμού θερμοδυναμικών ιδιοτήτων**…… 4-14
4.5.4.1 **Υποπρόγραμμα total**…… 4-14
4.5.4.2 **Υποπρόγραμμα energy**…… 4-14
4.5.4.3 **Υποπρόγραμμα enthalpy**…… 4-14
4.5.4.4 **Υποπρόγραμμα entropy** …….. 4-14
4.5.4.5 **Υποπρόγραμμα shp**…… .. 4-14
4.5.4.6 **Υποπρόγραμμα shv** …… 4-14
4.5.4.7 **Υποπρόγραμμα shr_sove** …… 4-15
4.5.4.8 **Υποπρόγραμμα ibm_jtc** …….. 4-15
4.5.5 **Υποπρογράμματα υπολογισμού ιδιοτήτων μεταφοράς** …… 4-15
4.5.5.1 **Υποπρόγραμμα viscosity** ……..... 4-15
4.5.5.2 **Υποπρόγραμμα conductive** ……..... 4-15
4.5.5.3 **Υποπρόγραμμα tension** ……..... 4-15
4.5.6 **Υποπρογράμματα μαθηματικών υπολογισμών** …… 4-15
4.5.7 **Άλλα υποπρογράμματα** ……..... 4-16
4.5.7.1 **Script initial** …… 4-16
4.5.7.2 **Script constants** …….. 4-16
4.5.7.3 **Script transient** ……..... 4-16
4.5.7.4 **Κυρίως πρόγραμμα example και συγκριτικά αποτελέσματα** ……..... 4-17
4.6 **Λειτουργία κωδίκων** …… 4-17
4.7 **Σχόλια και συμπεράσματα** …… 4-23
Πίνακες του 4^{ου} Κεφαλαίου..... 4-25

ΚΕΦΑΛΑΙΟ 5: **ΚΩΔΙΚΕΣ ΥΠΟΛΟΓΙΣΜΟΥ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB & SCILAB**

5.1 **Εισαγωγή**..... 5-1
5.2 **Λεπτομερής παρουσίαση του κώδικα**..... 5-2
5.2.1 **Το υποπρόγραμμα light_wasp (ή heavy_wasp)**..... 5-2
5.2.2 **Το υποπρόγραμμα checkpt** 5-4

5.2.3 **Το υποπρόγραμμα qmust**..... 5-5
5.2.4 **Το υποπρόγραμμα total**..... 5-6
5.2.5 **Τα υπόλοιπα υποπρογράμματα**..... 5-7
5.2.5.1 **Το υποπρόγραμμα fts**..... 5-7
5.2.5.2 **Το υποπρόγραμμα fps**..... 5-8
5.2.5.3 **Το υποπρόγραμμα dfps**..... 5-9
5.2.5.4 **Το υποπρόγραμμα densf**..... 5-9
5.2.5.5 **Το υποπρόγραμμα densg**..... 5-10
5.2.5.6 **Το υποπρόγραμμα svlwl**..... 5-10
5.2.5.7 **Το υποπρόγραμμα svlwv**..... 5-11
5.2.5.8 **Το υποπρόγραμμα prl**..... 5-13
5.2.5.9 **Το υποπρόγραμμα fpr3**..... 5-13
5.2.5.10 **Το υποπρόγραμμα dfpr3**..... 5-14
5.2.5.11 **Το υποπρόγραμμα fpr4**..... 5-15
5.2.5.12 **Το υποπρόγραμμα dfpr4**..... 5-16
5.2.5.13 **Το υποπρόγραμμα fp**..... 5-17
5.2.5.14 **Το υποπρόγραμμα dfpd**..... 5-18
5.2.5.15 **Το υποπρόγραμμα newton**..... 5-19
5.2.5.16 **Το υποπρόγραμμα energy**..... 5-21
5.2.5.17 **Το υποπρόγραμμα enthalpy**..... 5-22
5.2.5.18 **Το υποπρόγραμμα entropy**..... 5-23
5.2.5.19 **Το υποπρόγραμμα shp**..... 5-24
5.2.5.20 **Το υποπρόγραμμα shv**..... 5-25
5.2.5.21 **Το υποπρόγραμμα shr_sove**..... 5-25
5.2.5.22 **Το υποπρόγραμμα ibm_jtc**..... 5-26
5.2.5.23 **Το υποπρόγραμμα viscosity**..... 5-27
5.2.5.24 **Το υποπρόγραμμα conductive**..... 5-29
5.2.5.25 **Το υποπρόγραμμα tension**..... 5-31
5.3 **Σχόλια και συμπεράσματα**..... 5-31

ΚΕΦΑΛΑΙΟ 6: **ΑΝΤΙΚΑΤΑΣΤΑΣΗ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΩΝ ΕΡΓΑΛΕΙΩΝ FORTRAN ΑΠΟ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΑ ΕΡΓΑΛΕΙΑ MATLAB & SCILAB**

6.1 **Εισαγωγή**..... 6-1
6.2 **Κυρίως πρόγραμμα**..... 6-1

6.3 **Υποπρογράμματα**……..... 6-2
6.4 **Εντολές GOTO**……..... 6-3
6.5 **Μεταβλητές**…….. 6-15
6.6 **Τιμές στοιχείων πινάκων**……..... 6-17
6.7 **Εύρεση ρίζας εξίσωσης**……..... 6-18
6.8 **Διαδικασίες stop, return, end ή ισοδύναμες**…….. 6-19
6.9 **Δημιουργία εκτελέσιμου**……..... 6-20
6.10 **Διαδικασίες debugging**……. 6-21
6.11 **Σχόλια και συμπεράσματα**…….. 6-22

ΕΠΙΛΟΓΟΣ Ε-1

ΒΙΒΛΙΟΓΡΑΦΙΑ Β-1

ΠΑΡΑΡΤΗΜΑ..... Π-1

ΠΡΟΛΟΓΟΣ

Αυτή η Διπλωματική Εργασία αποτελεί συνέχεια της επιστημονικής προσπάθειας του Τομέα Πυρηνικής Τεχνολογίας του Ε.Μ.Π. σχετικά με τις ιδιότητες των Ψυκτικών, Εργαζόμενων και Επιβραδυντικών μέσων που χρησιμοποιούνται στους Πυρηνικούς Αντιδραστήρες. Ήδη από το 1987 έχει κατά ένα μεγάλο μέρος καλυφθεί σαν τέτοιο μέσο, με σειρά Διπλωματικών Εργασιών, το ελαφρύ νερό και από το 1986 ξεκίνησε η αντίστοιχη προσπάθεια για το βαρύ νερό με τη Διπλωματική Εργασία του Κωστάκου (1986), η οποία συνεχίσθηκε στα πλαίσια της Διπλωματικής Εργασίας του Πετρόπουλου (1991) καθώς και στα πλαίσια της Διδακτορικής Διατριβής του ίδιου (2003). Τις εργασίες αυτές ακολουθεί η παρούσα Διπλωματική Εργασία στην οποία γίνεται ανάπτυξη κωδίκων ηλεκτρονικού υπολογιστή για να αναπαράγουν με πολύ μεγάλη ακρίβεια θερμοδυναμικές ιδιότητες και ιδιότητες μεταφοράς του ελαφρού ύδατος, του βαρέος ύδατος, τα οποία είναι και τα συνηθέστερα χρησιμοποιούμενα σήμερα ψυκτικά μέσα στους Πυρηνικούς Αντιδραστήρες Ισχύος, σε προγραμματιστικό περιβάλλον MATLAB και SCILAB. Οι κώδικες δέχονται ως είσοδο τα καταστατικά θερμοδυναμικά μεγέθη πιέσεως και θερμοκρασίας και υπολογίζουν τα ειδικά μεγέθη όγκου, ενθαλπίας, εντροπίας, θερμοχωρητικότητας υπό σταθερή πίεση και υπό σταθερό όγκο και εσωτερικής ενέργειας καθώς και τις ιδιότητες μεταφοράς, θερμική αγωγιμότητα, δυναμική συνεκτικότητα και επιφανειακή τάση, τόσο για το ελαφρύ όσο και για το βαρύ ύδωρ. Οι κώδικες αναπτύχθηκαν με τέτοιο τρόπο ώστε να καλούνται εύκολα από το κυρίως πρόγραμμα του χρήστη, να επεκτείνονται επίσης εύκολα όταν χρειάζεται τόσο ώστε να υπολογίζονται και άλλες ιδιότητες όσο και να τροποποιούνται για να δέχονται σαν είσοδο άλλα θερμοδυναμικά μεγέθη, όπως για παράδειγμα την ενθαλπία και την πίεση ή την εντροπία και τη θερμοκρασία ή την ενθαλπία και την εντροπία. Η δομή των κωδίκων είναι τέτοια ώστε να είναι δυνατόν ο χρήστης να εκμεταλλεύεται μόνο εκείνα τα τμήματά τους που είναι απαραίτητα για τους υπολογισμούς του. Στο κείμενο της Διπλωματικής Εργασίας συμπεριλαμβάνονται λεπτομερείς οδηγίες χρήσης μαζί με πινακοποιημένες περιλήψεις που χρησιμεύουν στην ευκολότερη αξιοποίησή τους.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα Λέκτορα κ. Ν.Π. Πετρόπουλο για την συνεργασία που είχαμε καθ' όλη τη διάρκεια της εκπόνησης αυτής της Διπλωματικής Εργασίας.

ΠΕΡΙΛΗΨΗ

ΚΩΔΙΚΕΣ ΥΠΟΛΟΓΙΣΜΩΝ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB 7.8 (R2009a) ΚΑΙ SCILAB 5.2.2

Διπλωματική Εργασία του Σ. Παχίτσα

Οι κύριοι στόχοι της παρούσας Διπλωματικής Εργασίας συνοψίζονται ως εξής:

(α) στην εισαγωγική ανάπτυξη και πρώτη γνωριμία για τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, δεδομένου ότι τα περιβάλλοντα αυτά δεν έχουν ξαναχρησιμοποιηθεί για ανάπτυξη κωδίκων στον Τομέα Πυρηνικής Τεχνολογίας της Σχολής Μηχανολόγων του ΕΜΠ.

(β) στην επισκόπηση των εξισώσεων και των συσχετίσεων για τις θερμοφυσικές ιδιότητες ελαφρού και βαρέος ύδατος, όπως αυτές έχουν χρησιμοποιηθεί σε κώδικες υπολογισμού ιδιοτήτων που έχουν υλοποιηθεί σε γλώσσα FORTRAN, στα πλαίσια παλαιότερων Διπλωματικών Εργασιών του Τομέα Πυρηνικής Τεχνολογίας.

(γ) στην ανάπτυξη κωδίκων υπολογισμού θερμοφυσικών ιδιοτήτων ελαφρού και βαρέος ύδατος σε περιβάλλον MATLAB και SCILAB με βάση τη δομή και τη σύνταξη των αντίστοιχων κωδίκων που έχουν υλοποιηθεί σε γλώσσα FORTRAN. Οι κώδικες αυτοί πρέπει να λειτουργούν ώστε να δέχονται ως μεταβλητές εισόδου την πίεση και τη θερμοκρασία. Οι ιδιότητες που θα γίνονται διαθέσιμες σε κάθε υπολογισμό ως αποτελέσματα, είναι: πίεση, θερμοκρασία, πυκνότητα, εντροπία, ενθαλπία, ειδικές θερμοχωρητικότητες, ταχύτητα του ήχου, συνεκτικότητα, θερμική αγωγιμότητα, επιφανειακή τάση και σταθερά Laplace.

(δ) στην εκτενή περιγραφή της δομής και των οδηγιών χρήσης των κωδίκων υπολογισμού θερμοφυσικών ιδιοτήτων ελαφρού και βαρέος ύδατος σε περιβάλλον MATLAB και SCILAB με συνακόλουθη λεπτομερή εξέταση όλων των υποπρογραμμάτων κατά ομάδες ανάλογα και με τις λειτουργίες που αυτά διεκπεραιώνουν και της ροής της εκτέλεσης ανάλογα και με τα ζητούμενα που καθορίζει ο χρήστης.

(στ) στην ανάλυση σχετικά με τις διάφορες ιδιαιτερότητες του προγραμματισμού σε MATLAB και SCILAB σε σχέση και με το περιβάλλον προγραμματισμού FORTRAN. Συγκεκριμένα, χρειάζεται να παρουσιασθεί ο τρόπος με τον οποίο αντικαταστάθηκαν ορισμένα από τα προγραμματιστικά εργαλεία που υπάρχουν στη γλώσσα FORTRAN από αντίστοιχα προγραμματιστικά εργαλεία που διατίθενται στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB. Η παρουσίαση αυτή είναι απαραίτητη διότι ορισμένες από τις εντολές και τα εργαλεία της FORTRAN υλοποιούνται κάπως ή πολύ διαφορετικά στο MATLAB ή το SCILAB.

Στην πορεία της Διπλωματικής Εργασίας πιστεύεται ότι αντιμετωπίσθηκαν με επιτυχία όλοι οι πιοπάνω στόχοι και αποδόθηκε ως τελικό προϊόν οι κώδικες σε περιβάλλοντα προγραμματισμού MATLAB και SCILAB για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος για διάφορες θερμοκρασίες και πιέσεις που καλύπτουν την περιοχή ισχύος των χρησιμοποιούμενων σχετικών εξισώσεων. Διαπιστώθηκε ότι τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, πράγματι εξαιρετικά φιλικά προς τον χρήστη, διότι διαθέτουν μαθηματικά και γραφικά εργαλεία σε πολύ μεγαλύτερη έκταση από οποιοδήποτε άλλο σχεδόν προγραμματιστικό περιβάλλον. Η δοκιμή δημιουργίας κωδίκων θερμοφυσικών ιδιοτήτων σε αυτά τα περιβάλλοντα έπρεπε να γίνει για να διαπιστωθεί η χρηστικότητα τους και για να πραγματοποιηθεί στο μέλλον η εκμετάλλευση ορισμένων επιπλέον δυνατοτήτων που δίνονται από το MATLAB και το SCILAB, σε σύγκριση με τις δυνατότητες κωδίκων που έχουν γραφτεί σε γλώσσα FORTRAN.

ABSTRACT

COMPUTER CODES FOR THE CALCULATION OF THE THERMOPHYSICAL PROPERTIES OF LIGHT AND HEAVY WATER SUBSTANCES IN MATLAB 7.8 (R2009A) AND SCILAB 5.2.2 PROGRAMMING ENVIRONMENTS

Diploma Dissertation of S. Pachitsas

The main objectives of this Diploma Dissertation may be summarized as it follows:

(1) to briefly review and in summary present the MATLAB and SCILAB programming environments, since programming in them has not yet been applied in code development at the Nuclear Engineering Department of the National Technical University of Athens.

(2) to review the state equations and the relevant correlations for the calculation of the thermophysical properties of light (ordinary) and heavy water, that have been implemented in codes for propertied calculation developed in FORTRAN, in the course of Nuclear Engineering Department Diploma Dissertations as completed in the recent past.

(3) to develop codes for the calculation of the thermophysical properties of light and heavy water in MATLAB and SCILAB programming environments, based on the structure and the syntax of respective codes, which have been previously prepared in FORTRAN. The codes should work so that they could accept as input pressure and temperature. The properties, which should become available as results are pressure, temperature, density, entropy, enthalpy, heat capacities, speed of sound, viscosity, thermal conductivity, surface tension and the Laplace constant.

(4) to extensively describe the structure and use instructions of the codes developed for the calculation of the thermophysical properties of light and heavy water in the MATLAB and SCILAB programming environments. In this context, all classes of subprograms are to be reviewed in detail regarding their internal operation and also in conjunction with code execution flow and depending on which properties are requested by the user.

(6) to analyze certain peculiarities and differences between programming in MATLAB and SCILAB and programming in FORTRAN. In particular, there is a

specific need to present the way in which some FORTRAN programming tools and commands have been replaced by the respective tools in MATLAB and SCILAB. Such a presentation is necessary, since some of the commands and programming techniques used in FORTRAN are quite differently implemented in MATLAB and SCILAB.

It is believed that during the course of this Diploma Dissertation, all the above objectives, have been dealt with successfully. The final outcome was reliable codes for the calculation of the thermophysical properties of light and heavy water in the MATLAB and SCILAB programming environments for pressures and temperatures that cover the validity region of the implemented relevant equations and correlations. Moreover, it has been established that the MATLAB and SCILAB programming environments are actually very user-friendly, since they include extended mathematical and graphical tools which are missing from almost any other programming environment. The test development of these codes for the thermophysical properties in MATLAB and SCILAB was a necessary task, so that one could get familiar with their capabilities and in order to create a momentum for further exploiting the enhanced tools of MATLAB and SCILAB in the future.

ΚΕΦΑΛΑΙΟ 1
ΕΙΣΑΓΩΓΗ

1.1 Γενικά

Τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB ανήκουν στην λεγόμενη τέταρτη γενιά γλωσσών προγραμματισμού (συντομογραφία 4GL), της οποίας η περίοδος προσδιορισμού των χαρακτηριστικών της ανάγεται στην εικοσαετία 1970-1990. Η ανάπτυξη της 4ης γενιάς γλωσσών προγραμματισμού, επέτρεψε απλούστερο τρόπο προγραμματισμού, την ανάπτυξη λογισμικού με καλύτερους ρυθμούς, με αυξημένη αξιοπιστία και μικρότερο κόστος. Τα τελευταία χρόνια (1995 – 2010) η ευρεία διάδοση των περιβαλλόντων 4GL επιβοηθήθηκε τόσο από το καλύτερο διαθέσιμο υλικό (hardware) όσο και από τα νεότερα λειτουργικά συστήματα, που εισήγαγαν σημαντικές διευκολύνσεις. Έτσι, ο όρος "γλώσσα προγραμματισμού 4GL" ή "περιβάλλον προγραμματισμού 4GL" περιγράφει σήμερα (2010) μια μεγάλη σειρά προϊόντων λογισμικού που εξαιτίας του πολύ υψηλού επιπέδου επικοινωνίας τους με τον χρήστη δεν απευθύνονται στο στενό κοινό των προγραμματιστών αλλά σε ένα πολύ ευρύτερο κοινό καταναλωτών. Κατά μία έννοια, τα προϊόντα 4GL είναι πλέον κλασικές εφαρμογές "black – box", δεδομένου ότι η σύνταξη και η εκτέλεση του προγράμματος εξαρτάται πολύ λιγότερο από το υλικό, στο οποίο αυτό θα λειτουργήσει, από τα αντίστοιχα προϊόντα 3GL. Τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB ως εργαλεία προγραμματισμού 4GL είναι πολύ φιλικά για τον χρήστη. Σε επίπεδο εντολών μπορούν να υλοποιήσουν κώδικες με τρόπο τουλάχιστον ανάλογο με τον τρόπο που γίνεται αυτό στη FORTRAN. Οι δυνατότητές τους επομένως είναι κατάλληλες για την υλοποίηση και κωδίκων για τον υπολογισμό θερμοφυσικών ιδιοτήτων εργαζόμενων μέσων, αρκεί να δίνονται οι κατάλληλες καταστατικές εξισώσεις και οι εξισώσεις ιδιοτήτων μεταφοράς. Στο $2^{ο}$ Κεφαλαίο της παρούσης Διπλωματικής Εργασίας γίνεται μία εισαγωγική ανάπτυξη και πρώτη γνωριμία για τα περιβάλλοντα MATLAB και SCILAB.

1.2 Εξισώσεις για τις θερμοφυσικές ιδιότητες ελαφρού και βαρέος ύδατος

Στο 3^{o} Κεφάλαιο της Διπλωματικής Εργασίας, δίνονται τα βασικά στοιχεία των καταστατικών εξισώσεων και των εξισώσεων ιδιοτήτων για δύο από τα σημαντικότερα ψυκτικά μέσα στην τεχνολογία των Πυρηνικών Αντιδραστήρων Ισχύος, το ελαφρύ και το βαρύ νερό, για τα οποία αναπτύχθηκαν κώδικες υπολογισμού θερμοφυσικών ιδιοτήτων σε περιβάλλον MATLAB και SCILAB. Φυσικά, τυποποιημένοι πίνακες θερμοφυσικών ιδιοτήτων (δηλ. θερμοδυναμικών και ιδιοτήτων μεταφοράς) ελαφρού νερού και ατμού είναι διαθέσιμοι στους μηχανικούς από το πρώτο μισό του περασμένου αιώνα. Το βαρύ νερό έχει παρόμοιες θερμοφυσικές ιδιότητες με το ελαφρύ παρόλαυτά σε αντίθεση με το ελαφρύ, δεν είναι φθηνό ούτε άμεσα διαθέσιμο. Επιπλέον, τυποποιημένοι πίνακες θερμοφυσικών ιδιοτήτων βαρέος ύδατος και ατμού έγιναν διαθέσιμοι στους μηχανικούς μόλις περί το τέλος του περασμένου αιώνα. Τόσο για το ελαφρύ όσο και για το βαρύ νερό διατίθενται υπολογιστικοί κώδικες που συνήθως παρεμβάλλουν σε αυτούς τους πίνακες χρησιμοποιώντας μία ποικιλία καμπυλών και μεθόδων παρεμβολής. Πολλοί από αυτούς τους κώδικες είναι αργοί και παρουσιάζουν έλλειψη αξιοπιστίας αποτελεσμάτων ενώ κάποιοι άλλοι έχουν σχεδιαστεί για εξειδικευμένες χρήσεις και δεν παρέχουν πολλές από τις θερμοφυσικές ιδιότητες που ενδιαφέρουν. Οι εξισώσεις ιδιοτήτων, όπως παρουσιάζονται στο 3^{o} Κεφάλαιο, για τις περιοχές πιέσεων και θερμοκρασιών για τις οποίες ισχύουν, μπορούν να υλοποιηθούν σε κώδικες σε περιβάλλον MATLAB ή SCILAB, για τον υπολογισμό των θερμοδυναμικών ιδιοτήτων και των ιδιοτήτων μεταφοράς του ελαφρού και του βαρέος ύδατος και των αντίστοιχων ατμών. Οι κώδικες αυτοί σχεδιάσθηκαν να δέχονται ως μεταβλητές εισόδου την πίεση (σε bar) και τη θερμοκρασία (σε ^{o}C). Οι ιδιότητες που γίνονται διαθέσιμες σε κάθε υπολογισμό ως αποτελέσματα (έξοδοι), είναι: πίεση, θερμοκρασία, πυκνότητα, εντροπία, ενθαλπία, ειδικές θερμοχωρητικότητες, ταχύτητα του ήχου, συνεκτικότητα, θερμική αγωγιμότητα, επιφανειακή τάση και σταθερά Laplace. Οι κώδικες κατασκευάσθηκαν ώστε να αποτελούνται από επιμέρους τμήματα (functions) που επιλέγει να τα καλέσει ο χρήστης ανάλογα με το αν είναι απαραίτητα για τους υπολογισμούς του και είναι σχεδιασμένοι να λειτουργούν σαν συνάρτηση εντός του κύριου προγράμματος του

χρήστη. Επιπλέον οι κώδικες έχουν την δυνατότητα επίτευξης υπολογισμών μετασταθών καταστάσεων.

1.3 Δομή κωδίκων

Στο 4^{ο} Κεφάλαιο της Διπλωματικής Εργασίας, δίνονται λεπτομέρειες σχετικά με τις οδηγίες χρήσης των κωδίκων αυτών. Για το σκοπό αυτό γίνεται η παρουσίαση των αριθμητικών κωδίκων με τα ονόματα light_wasp (light water and steam properties) και heavy_wasp (heavy water and steam properties), σε επίπεδο δομής και λογικής λειτουργίας. Οι κώδικες αυτοί βασίζονται σε αντίστοιχους που υλοποιήθηκαν παλαιότερα σε γλώσσα προγραμματισμού FORTRAN, στα πλαίσια της Διδακτορικής Διατριβής του Πετρόπουλου (2003), με σκοπό να εξυπηρετήσουν υπολογισμούς για τις θερμοδυναμικές ιδιότητες και τις ιδιότητες μεταφοράς του ελαφρού ύδατος και του βαρέος ύδατος αντίστοιχα. Το επίπεδο ακρίβειας των υπολογισμών είναι κατάλληλο για βιομηχανική χρήση, προκειμένου να χρησιμοποιηθεί για υπολογισμούς σε προβλήματα θερμοϋδραυλικής ανάλυσης Πυρηνικών Αντιδραστήρων Ισχύος. Οι οδηγίες χρήσεως συμπληρώνονται με ορισμένες υποδείξεις σχετικά με τους τρόπους με τους οποίους ο χρήστης μπορεί να αντιμετωπίσει τα συνηθέστερα λάθη των κωδίκων. Όλα τα υποπρογράμματα εξετάζονται στη συνέχεια κατά ομάδες ανάλογα και με τις λειτουργίες που αυτά διεκπεραιώνουν και παρουσιάζεται η ροή της εκτέλεσης ανάλογα και με τα ζητούμενα που καθορίζει ο χρήστης. Η συνολική παρουσίαση συνοδεύεται από συγκεντρωτικούς πίνακες που χρησιμεύουν ως περιληπτική αναφορά στα κυριότερα σημεία των κωδίκων. Παρέχονται φυσικά και τις οδηγίες σχετικά με τη σύνδεση ενός κυρίου προγράμματος με τον κώδικα light_wasp (ή heavy_wasp).

1.4 Λειτουργία υποπρογραμμάτων

Στο 5^{ο} Κεφάλαιο της Διπλωματικής Εργασίας, παρουσιάζονται οι κώδικες light_wasp και heavy_wasp αντίστοιχα με περισσότερη λεπτομέρεια, σε επίπεδο περιγραφής της εσωτερικής λειτουργίας των υποπρογραμμάτων στο περιβάλλον προγραμματισμού MATLAB (έκδοση R2009a) και SCILAB (έκδοση 5.2.2) δεδομένου ότι οι δύο κώδικες μοιράζονται σημαντικά κοινά μέρη και ότι τα δύο περιβάλλοντα προγραμματισμού είναι παρόμοια. Ιδιαίτερη έμφαση δίνεται στην εξέταση των υποπρογραμμάτων light_wasp (ή heavy_wasp), checkpt, qmust και total, τα οποία είναι τα μεγαλύτερα και σημαντικότερα

υποπρογράμματα. Τα υπόλοιπα υποπρογράμματα παρουσιάζονται συνοπτικότερα. Το σύνολο των υποπρογραμμάτων παρουσιάζεται κατά το δυνατόν με τη σειρά που χρησιμοποιούνται στον κώδικα. Οι κυριότερες συνιστώσες της παρουσίασης κάθε υποπρογράμματος είναι (α) η εξίσωση ή οι εξισώσεις οι οποίες υλοποιούνται (β) Μεταβλητές εισόδου, (γ) Μεταβλητές εξόδου, (δ) Τοπικές μεταβλητές, (ε) οι μονάδες που έχουν ορισμένες από τις μεταβλητές, (στ) Υποπρογράμματα που καλούν το παρουσιαζόμενο υποπρόγραμμα και (ζ) Υποπρογράμματα που καλεί το παρουσιαζόμενο υποπρόγραμμα.

1.5 Προγραμματιστικά εργαλεία MATLAB και SCILAB

Στο 6^{ο} Κεφάλαιο της Διπλωματικής Εργασίας γίνονται αναγκαίες παρατηρήσεις σχετικά με τις διάφορες ιδιαιτερότητες του προγραμματισμού σε MATLAB και SCILAB σε σχέση και με το περιβάλλον προγραμματισμού FORTRAN. Συγκεκριμένα, παρουσιάζεται ο τρόπος με τον οποίο αντικαταστάθηκαν ορισμένα από τα προγραμματιστικά εργαλεία που υπάρχουν στη γλώσσα FORTRAN από αντίστοιχα προγραμματιστικά εργαλεία που διατίθενται στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB. Η παρουσίαση αυτή είναι απαραίτητη διότι ορισμένες από τις εντολές και τα εργαλεία της FORTRAN υλοποιούνται κάπως ή πολύ διαφορετικά στο MATLAB ή το SCILAB. Ειδικά, σημαντικές διαφορές υπάρχουν στην έννοια του κυρίως προγράμματος, στον τύπο και τη λειτουργία των διαθέσιμων υποπρογραμμάτων, στην μεταβίβαση του ελέγχου μέσω GOTO, στον τρόπο που καθορίζονται και αλλάζουν τιμές οι τοπικές και οι καθολικές μεταβλητές, στον τρόπο που μπορούν να προσδιορίζονται οι ρίζες εξίσωσης, στον τρόπο που λειτουργούν οι εντολές stop, return, end ή ισοδύναμες, στον τρόπο που παράγεται εκτελέσιμος κώδικας και τέλος στις διαδικασίες debugging. Η παρουσίαση γίνεται σε διακριτές παραγράφους για κάθε μία από τις σημαντικές διαφορές που αναφέρθηκαν.

1.6 Συμπεράσματα

Στην παρούσα Διπλωματική Εργασία αναπτύχθηκαν με επιτυχία κώδικες σε περιβάλλοντα προγραμματισμού MATLAB και SCILAB για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος για διάφορες θερμοκρασίες και πιέσεις που καλύπτουν την περιοχή ισχύος των χρησιμοποιούμενων

σχετικών εξισώσεων. Διαπιστώθηκε ότι τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, πράγματι εξαιρετικά φιλικά προς τον χρήστη, διότι διαθέτουν μαθηματικά και γραφικά εργαλεία σε πολύ μεγαλύτερη έκταση από οποιοδήποτε άλλο σχεδόν προγραμματιστικό περιβάλλον. Επιπλέον ειδικά το SCILAB παρέχεται χωρίς κόστος, πράγμα που αντισταθμίζει την περιορισμένη του επέκταση, ορισμένες λιγότερες ευκολίες καθώς επίσης και την περιορισμένη τεκμηρίωση σε σχέση με το MATLAB. Επομένως η δοκιμή δημιουργίας κωδίκων θερμοφυσικών ιδιοτήτων σε αυτά τα περιβάλλοντα έπρεπε να γίνει για να διαπιστωθεί η χρηστικότητα τους και για να πραγματοποιηθεί στο μέλλον η εκμετάλλευση ορισμένων επιπλέον δυνατοτήτων που δίνονται από το MATLAB και το SCILAB, σε σύγκριση με τις δυνατότητες κωδίκων που έχουν γραφτεί σε γλώσσα FORTRAN.

1.7 Παράρτημα

Ως συνέχεια του 4^{ου} και του 5^{ου} Κεφαλαίου σε Παράρτημα παρουσιάζονται τα κείμενα των υποπρογραμμάτων των κωδίκων, καθώς και πίνακες με ενδεικτικά αποτελέσματα.

ΚΕΦΑΛΑΙΟ 2
ΣΥΝΤΟΜΗ ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΠΕΡΙΒΑΛΛΟΝΤΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB ΚΑΙ SCILAB

2.1 Εισαγωγή

Τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB ανήκουν στην λεγόμενη τέταρτη γενιά γλωσσών προγραμματισμού (συντομογραφία 4GL), της οποίας η περίοδος προσδιορισμού των χαρακτηριστικών της ανάγεται στην εικοσαετία 1970-1990. Στην ιστορία της επιστήμης των υπολογιστών, η 4GL ακολούθησε την 3GL με κύριο στόχο τη διάθεση στο χρήστη απλούστερων και πολύ δυνατότερων εντολών. Στις γλώσσες προγραμματισμού τρίτης γενιάς υπήρχε η δυνατότητα δομημένων λειτουργιών, και διευκολύνθηκε έτσι η ανάπτυξη λογισμικού για πολλές ανάγκες. Παρόλαυτά σε αυτές τις γλώσσες διαπιστώθηκε χαμηλός ρυθμός ανάπτυξης του λογισμικού και ισχυρή πιθανότητα προγραμματισμού με λάθη. Έτσι παρουσιάστηκε η ανάγκη ανάπτυξης της 4ης γενιάς γλωσσών προγραμματισμού, που να επιτρέπει απλούστερο τρόπο προγραμματισμού, την ανάπτυξη λογισμικού με καλύτερους ρυθμούς, με αυξημένη αξιοπιστία και μικρότερο κόστος. Ο όρος 4GL χρησιμοποιήθηκε για πρώτη φορά επισήμως από τον James Martin, το 1982, στο βιβλίο του "Applications Development Without Programmers". Η γλώσσα RPG της IBM (1960) θα μπορούσε να περιγραφεί ως η πρώτη 4GL (σε πρωτόγονη μορφή βέβαια) και ακολουθήσαν και άλλες όπως η Informatics MARK-IV (1967) και η MAPPER της Sperry (που δόθηκε στην αγορά το 1979). Η ανάπτυξη των περιβαλλόντων 4GL επηρεάστηκε τόσο από το διαθέσιμο υλικό (hardware) όσο και από τα διαθέσιμα λειτουργικά συστήματα, που εισήγαγαν σημαντικούς περιορισμούς. Συγκεκριμένα, όταν εμφανίσθηκαν τα πρώτα περιβάλλοντα 4GL αυτά ήταν προορισμένα μόνο για ένα υλικό και ένα λειτουργικό σύστημα. Χαρακτηριστικό παράδειγμα είναι το σύστημα MAPPER που αναπτύχθηκε από την Sperry. Οι μετέπειτα τύποι περιβαλλόντων 4GL συνδέθηκαν με βάσεις δεδομένων και έτσι διαφοροποιήθηκαν αρκετά από τα πρώιμα προϊόντα ως προς τη χρήση των τεχνικών μέσων και των πόρων του λειτουργικού συστήματος. Ο όρος "γλώσσα προγραμματισμού 4GL" ή "περιβάλλον προγραμματισμού 4GL" περιγράφει σήμερα (2010) μια μεγάλη σειρά προϊόντων λογισμικού που εξαιτίας του πολύ υψηλού επιπέδου επικοινωνίας τους με τον χρήστη δεν απευθύνονται στο στενό κοινό των προγραμματιστών αλλά σε ένα πολύ ευρύτερο κοινό καταναλωτών. Κατά μία έννοια,

τα προϊόντα 4GL είναι πλέον κλασικές εφαρμογές "black – box", δεδομένου ότι η σύνταξη και η εκτέλεση του προγράμματος εξαρτάται πολύ λιγότερο από το υλικό, στο οποίο αυτό θα λειτουργήσει, από τα αντίστοιχα προϊόντα 3GL.

2.2 Βασικά στοιχεία του MATLAB

Η ονοματολογία του προγραμματιστικού περιβάλλοντος MATLAB προέρχεται από συνδυασμό των λέξεων Matrix και Lab. Πρόκειται, όπως σημειώθηκε, για ένα αριθμητικό - υπολογιστικό περιβάλλον τέταρτης γενιάς. Αναπτύχθηκε από την MathWorks ως περιβάλλον εναλλακτικό της γλώσσας προγραμματισμού FORTRAN και επιτρέπει επιπλέον εύκολους χειρισμούς πινάκων, παραγωγή γραφημάτων από συναρτήσεις ή δεδομένα, υλοποίηση πολύπλοκων αλγορίθμων και εργασία σε πολύ φιλικό περιβάλλον για τον χρήστη. Επίσης μπορεί, υπό προϋποθέσεις, να συνδεθεί με προγράμματα γραμμένα σε άλλες γλώσσες προγραμματισμού, συμπεριλαμβανομένης της C, της C + + και της Fortran. Είναι εμπορικά διαθέσιμο για λειτουργικά συστήματα Windows, Linux και MAC OS-X σε Intel (32 και 64 bit). Παρά το γεγονός ότι το MATLAB προορίζεται κυρίως για την αριθμητικούς υπολογισμούς, διατίθενται εργαλεία που το μετατρέπουν σε περιβάλλον συμβολικού προγραμματισμού που παρέχουν δυνατότητες συμβολικών υπολογισμών. Το πρόσθετο πακέτο που συνεργάζεται με το MATLAB, το γνωστό Simulink, προσθέτει δυνατότητες γραφικής προσομοίωσης επεξεργασίας δεδομένων και παραγωγής αποτελεσμάτων. Το 2004, η MathWorks υποστήριξε ότι, το MATLAB ξεπέρασε τους ένα εκατομμύριο χρήστες σε ολόκληρο το κόσμο. Οι χρήστες του MATLAB προέρχονται από διάφορους κλάδους της μηχανικής, των επιστημών, και της οικονομίας. Μεταξύ αυτών βρίσκονται τα περισσότερα ακαδημαϊκά και ερευνητικά ιδρύματα.

Ο απλούστερος τρόπος για να εκτελεστεί ένας κώδικας σε MATLAB είναι να πληκτρολογηθεί το όνομα του στη "γραμμή εντολών" του "παραθύρου εντολών" (ή Command Window). Τόσο η γραμμή εντολών όσο και το παράθυρο εντολών είναι βασικό στοιχείο της επιφάνειας εργασίας (ή Desktop) του MATLAB. Με τον τρόπο αυτό, το MATLAB μπορεί να χρησιμοποιηθεί ως διαδραστικό μαθηματικό κέλυφος προγραμματισμού. Αλληλουχίες εντολών μπορούν να αποθηκευτούν σε ένα αρχείο κειμένου, χρησιμοποιώντας τον διατιθέμενο επεξεργαστή κειμένου (ή Editor) του MATLAB, αλλά και όποιον άλλο επεξεργαστή διατίθεται. Προκειμένου το αρχείο αυτό να λογίζεται ως μέρος από κώδικα MATLAB πρέπει να έχει την επέκταση ".m".

Το MATLAB μπορεί, υπό ορισμένες προϋποθέσεις, να καλεί υποπρογράμματα που είναι γραμμένα σε άλλες γλώσσες προγραμματισμού όπως π.χ. η C και η FORTRAN. Για το σκοπό αυτό διατίθεται ένα ειδικό κέλυφος προγραμματισμού που επιτρέπει στο MATLAB να λαμβάνει και να επιστραφεί δεδομένα χρησιμοποιώντας προγράμματα που είναι γραμμένα σε αυτές τις γλώσσες. Τα αρχεία που δημιουργούνται για την εφαρμογή τέτοιων λειτουργιών ονομάζονται "MEX-files". Επιπλέον υπό ορισμένες προϋποθέσεις βιβλιοθήκες γραμμένες σε Java, ActiveX ή NET μπορούν να καλούνται από το MATLAB. Η κλήση κώδικα σε MATLAB από περιβάλλον προγραμματισμού Java γίνεται με κατάλληλη επέκταση λογισμικού που πωλείται ξεχωριστά από την MathWorks, είτε εναλλακτικά χρησιμοποιώντας ένα μηχανισμό που ονομάζεται JMI (Java-to-Matlab Interface).

Πρέπει να σημειωθεί εδώ ότι το περιβάλλον προγραμματισμού MATLAB, δεν παράγει εκτελέσιμο από μεταγλώττιση και σύνδεση (compilation and linking) αλλά εκτελεί τις εντολές με την διαδικασία διερμηνείας, είναι δηλαδή όπως λέγεται "interpreter".

2.3 Βασικά στοιχεία του SCILAB

Η ονοματολογία του προγραμματιστικού περιβάλλοντος SCILAB προέρχεται από συνδυασμό των λέξεων Scientific και Lab. Πρόκειται, όπως σημειώθηκε, για ένα αριθμητικό - υπολογιστικό περιβάλλον τέταρτης γενιάς. Είναι ένα περιβάλλον ανοικτού κώδικα (open source) ανεξάρτητο από λειτουργικό σύστημα (cross-platform) που παρέχει στον χρήστη μια υψηλού επιπέδου, γλώσσα αριθμητικού προγραμματισμού, το οποίο ανταγωνίζεται σε ποιότητα και δυνατότητες το περιβάλλον MATLAB. Είναι ελεύθερα διαθέσιμο για λειτουργικά συστήματα Windows, Linux και MAC OS-X σε Intel (32 και 64 bit). Τα συνηθισμένα πεδία εφαρμογών είναι: (α) η επεξεργασία σήματος, (β) η στατιστική ανάλυση, (γ) η επεξεργασία εικόνας, (δ) η αριθμητική βελτιστοποίηση, και (ε) η μοντελοποίηση - προσομοίωση. Το SCILAB δημιουργήθηκε το 1990 στη Γαλλία από ερευνητές του INRIA (του Γαλλικού Εθνικού Ινστιτούτου για την Έρευνα στους Ηλεκτρονικούς Υπολογιστές) και του Ecole Nationale des Ponts et Chaussées (ENPC). Η κοινοπραξία που διαθέτει σήμερα το SCILAB ιδρύθηκε το Μάιο του 2003 με σκοπό την προώθησή του σε όλο τον κόσμο ως λογισμικό αναφοράς στα πανεπιστήμια και τη βιομηχανία.

Για το περιβάλλον προγραμματισμού SCILAB οι πίνακες είναι κύριος τύπος μεταβλητών, δεδομένων και αποτελεσμάτων. Η χρησιμοποίηση πινάκων ως βάση υπολογισμών σε συνδυασμό με την αυτόματη διαχείριση μνήμης που τους αντιστοιχεί, επιτρέπει την έκφραση πολλών αριθμητικών προβλημάτων με πολύ μειωμένο αριθμό γραμμών κώδικα, συγκριτικά με παρόμοιες λύσεις στις παραδοσιακές γλώσσες, όπως οι FORTRAN, C, C++. Αυτό το πλεονέκτημα επιτρέπει στους χρήστες να κατασκευάσουν γρήγορα μοντέλα, για μια σειρά μαθηματικών προβλημάτων.

Ο απλούστερος τρόπος για να εκτελεστεί ένας κώδικας σε SCILAB είναι να πληκτρολογηθεί το όνομα του στη "γραμμή εντολών" του "παραθύρου εντολών" (ή Command Window). Όπως και στο MATLAB, η γραμμή εντολών και το παράθυρο εντολών είναι βασικό στοιχείο της επιφάνειας εργασίας (ή Desktop) του SCILAB. Με τον τρόπο αυτό, το SCILAB μπορεί να χρησιμοποιηθεί ομοίως ως διαδραστικό μαθηματικό κέλυφος προγραμματισμού. Αλληλουχίες εντολών μπορούν να αποθηκευτούν σε ένα αρχείο κειμένου, χρησιμοποιώντας τον διατιθέμενο επεξεργαστή κειμένου (ή Editor) του SCILAB, αλλά και όποιον άλλο επεξεργαστή διατίθεται. Προκειμένου το αρχείο αυτό να λογίζεται ως μέρος από κώδικα SCILAB πρέπει να έχει την επέκταση ".sci". Το SCILAB μπορεί, υπό ορισμένες προϋποθέσεις, να καλεί υποπρογράμματα που είναι γραμμένα σε άλλες γλώσσες προγραμματισμού όπως π.χ. η C και η FORTRAN.

Η σύνταξη κώδικα στο SCILAB είναι παρόμοια με το MATLAB με εξαίρεση την ύπαρξη κάποιων ελάχιστων ιδιαιτεροτήτων. Πρέπει να σημειωθεί και εδώ ότι το περιβάλλον προγραμματισμού SCILAB, δεν παράγει εκτελέσιμο από μεταγλώττιση και σύνδεση (compilation and linking) αλλά εκτελεί τις εντολές με την διαδικασία διερμηνείας, είναι δηλαδή όπως λέγεται "interpreter".

2.4 "Τρέξιμο" κώδικα σε MATLAB ή SCILAB

2.4.1 Γενικά

Στο σημείο αυτό παρουσιάζεται με παραδείγματα, ο τρόπος με τον οποίο ο χρήστης εισέρχεται σε περιβάλλον MATLAB και SCILAB με σκοπό να "τρέξει" ένα πρόγραμμα γραμμένο στην αντίστοιχη γλώσσα προγραμματισμού σε περιβάλλον Λειτουργικού Συστήματος MS Windows XP. Για τα παραδείγματα αυτά τα προγράμματα που θα τρέξουν είναι το example.m για το MATLAB και το example.sci για το SCILAB. Το καθένα από αυτά καλεί με την σειρά του άλλα

υποπρογράμματα που περιέχουν εντολές (script) και υποπρογράμματα (function). Η εκτέλεση του example(.m ή sci) επιτυγχάνει υπολογισμό θερμοφυσικών ιδιοτήτων βαρέος ύδατος για ένα ζεύγος τιμών πίεσης – θερμοκρασίας που δίνεται από τον χρήστη.

2.4.2 "Τρέξιμο" MATLAB

Το περιβάλλον MATLAB R2009a ενεργοποιείται επιλέγοντας από την λίστα των προγραμμάτων το εικονίδιο με το όνομα MATLAB R2009a, όπως φαίνεται στην εικόνα του Διαγράμματος 2.1. Ως αποτέλεσμα, εμφανίζεται στην οθόνη του υπολογιστή το περιβάλλον MATLAB, όπως παρουσιάζεται στο Διάγραμμα 2.2. Για την εκτέλεση προγράμματος example.m επιλέγεται από τη διαδρομή επιλογών Current Directory (βλ. αριστερά στο Διάγραμμα 2.2), ο φάκελος, στον οποίο εμπεριέχεται το συγκεκριμένο αρχείο. Εδώ πρόκειται π.χ. για τον φάκελο heavy_water_matlab-28-6-10. Με την επιλογή του συγκεκριμένου φακέλου παρουσιάζεται στο παράθυρο της διαδρομής επιλογών Current Directory το περιεχόμενο του και από εκεί επιλέγεται το example.m όπως φαίνεται στην εικόνα του Διαγράμματος 2.3. Μετά την επιλογή ενεργοποιείται ο Desktop Editor του MATLAB, όπως στην εικόνα του Διαγράμματος 2.4. Από τη διαδρομή επιλογών Debug/ Run ή από το πράσινο βέλος των εικονιδίων της γραμμής εργαλείων επιτυγχάνεται η εκτέλεση του προγράμματος. Με την έναρξη της εκτέλεσης του προγράμματος παρουσιάζεται η εικόνα του Διαγράμματος 2.5 (Command Window), όπου ζητείται η εισαγωγή τιμής πίεσης σε bar. Αφού δοθεί τιμή πίεσης και πατηθεί το πλήκτρο ENTER εμφανίζεται η εικόνα του Διαγράμματος 2.6 και ζητείται η εισαγωγή τιμής θερμοκρασίας σε ^{o}C. Με την τοποθέτηση της τιμής της θερμοκρασίας και μετά το πλήκτρο ENTER αρχίζει ο υπολογισμός των θερμοφυσικών ιδιοτήτων του βαρέος ύδατος για τις τιμές πίεσης και θερμοκρασίας που έχει δώσει ο χρήστης καθώς και η παρουσίαση των τιμών τους (βλ. πχ. και την εικόνα του Διαγράμματος 2.7).

Υπάρχουν και άλλοι δύο εναλλακτικοί τρόποι εκτέλεσης του προγράμματος example.m. Κατά τον πρώτο όταν ανοίξει ο φάκελος π.χ. heavy_water_matlab-28-6-10, στον οποίο εμπεριέχεται το πρόγραμμα example.m και γίνει η επιλογή του ονόματος του με "διπλό αριστερό κλικ", όπως φαίνεται στην εικόνα του Διαγράμματος 2.8 αυτόματα ανοίγει η εικόνα του Διαγράμματος 2.4. Έπειτα ακολουθείται η διαδικασία που έχει ήδη περιγραφτεί. Στον δεύτερο τρόπο μπορεί να

χρησιμοποιηθεί η εντολή "cd" στο Command Window για την αλλαγή φακέλου και είσοδο στον φάκελο που εμπεριέχει το πρόγραμμα που ενδιαφέρει. Έπειτα πληκτρολογείται το όνομα του προγράμματος στο Command Window και πατώντας το ENTER επιτυγχάνεται η εκτέλεση του προγράμματος. Λαμβάνεται έτσι μια εικόνα της μορφής του Διαγράμματος 2.5 και η διαδικασία συνεχίζεται όπως έχει περιγραφτεί.

2.4.3 "Τρέξιμο" SCILAB

Το περιβάλλον SCILAB-5.2.2 ενεργοποιείται επιλέγοντας από την λίστα των προγραμμάτων το εικονίδιο με το όνομα SCILAB-5.2.2, όπως φαίνεται στην εικόνα του Διαγράμματος 2.9. Μετά την ανωτέρω επιλογή εμφανίζεται στην οθόνη του υπολογιστή το περιβάλλον SCILAB όπως παρουσιάζεται στο Διάγραμμα 2.10. Η εκτέλεση προγράμματος example.sci επιτυγχάνεται με την ακόλουθη διαδικασία:

(α) επιλέγεται το εικονίδιο open όπως φαίνεται στην εικόνα του Διαγράμματος 2.11, (β) επιλέγεται ο φάκελος που περιέχει το πρόγραμμα που ενδιαφέρει, και στη συνέχεια το πρόγραμμα example.sci, με τον τρόπο που παρουσιάζεται στο Διάγραμμα 2.12. Μετά από αυτήν την επιλογή ενεργοποιείται ο Desktop Editor του SCILAB, όπως στην εικόνα του Διαγράμματος 2.13 και με την επιλογή του εικονιδίου "execute" όπως διακρίνεται σε αυτό γίνεται η εκτέλεση του προγράμματος.

Με την έναρξη της εκτέλεσης του προγράμματος παρουσιάζεται η εικόνα του Διαγράμματος 2.14 (Command Window), όπου ζητείται η δήλωση τιμής πίεσης σε bar. Αφού δοθεί τιμή πίεσης και πατηθεί το πλήκτρο ENTER εμφανίζεται η εικόνα του Διαγράμματος 2.15 και ζητείται τιμή θερμοκρασίας σε ^{o}C. Με την τοποθέτηση της τιμής της θερμοκρασίας και μετά το πλήκτρο ENTER αρχίζει ο υπολογισμός των θερμοφυσικών ιδιοτήτων του βαρέος ύδατος για τις τιμές πίεσης και θερμοκρασίας που έχει δώσει ο χρήστης καθώς και η παρουσίαση των τιμών τους όπως στην εικόνα του Διαγράμματος 2.16.

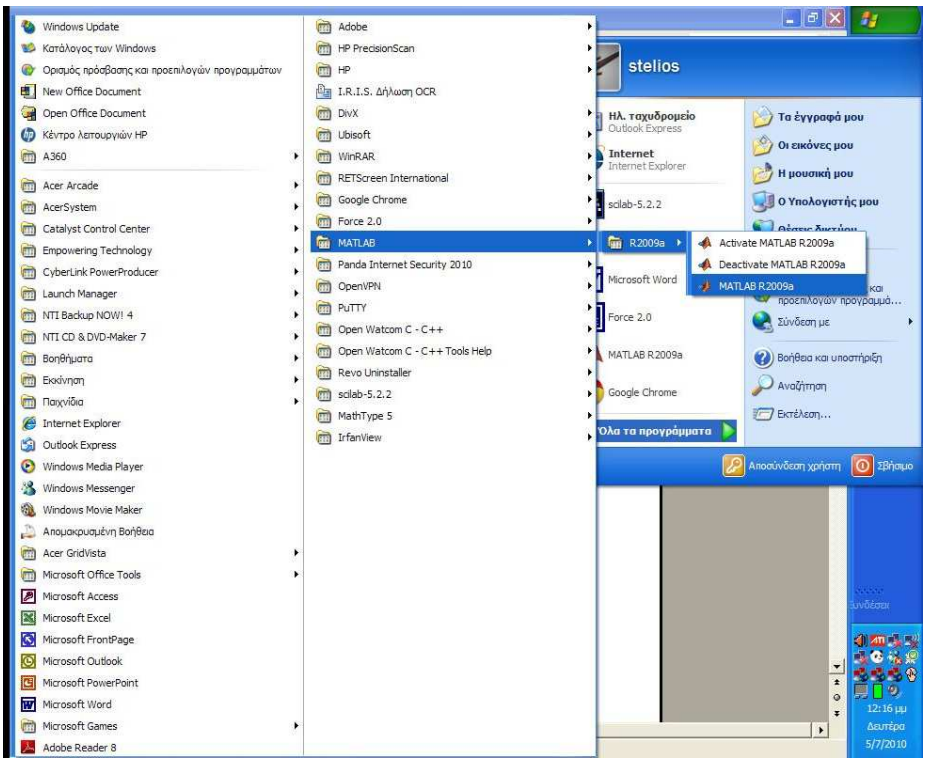
Υπάρχουν και άλλοι δυο εναλλακτικοί τρόποι εκτέλεσης του προγράμματος example.sci. Κατά τον πρώτο όταν ανοίξει ο φάκελος π.χ. scilab_hw, στον οποίο εμπεριέχεται το πρόγραμμα example.sci και γίνει η επιλογή του ονόματος του με "διπλό αριστερό κλικ", όπως φαίνεται στην εικόνα του Διαγράμματος 2.17 αυτόματα ανοίγει η εικόνα του Διαγράμματος 2.13. Έπειτα ακολουθείται η διαδικασία που έχει ήδη περιγραφεί. Στον δεύτερο τρόπο μπορεί να χρησιμοποιηθεί η εντολή "chdir" στο

Command Window για την αλλαγή φακέλου και είσοδο στο φάκελο που εμπεριέχει το πρόγραμμα που ενδιαφέρει. Έπειτα πληκτρολογείται η εντολή "exec" ακολουθούμενη με κενό και το όνομα του προγράμματος και πατώντας το ENTER επιτυγχάνεται η εκτέλεσή του. Λαμβάνεται έτσι μια εικόνα της μορφής του Διαγράμματος 2.14 και η διαδικασία συνεχίζεται όπως έχει περιγραφτεί.

2.5 Ανακεφαλαίωση

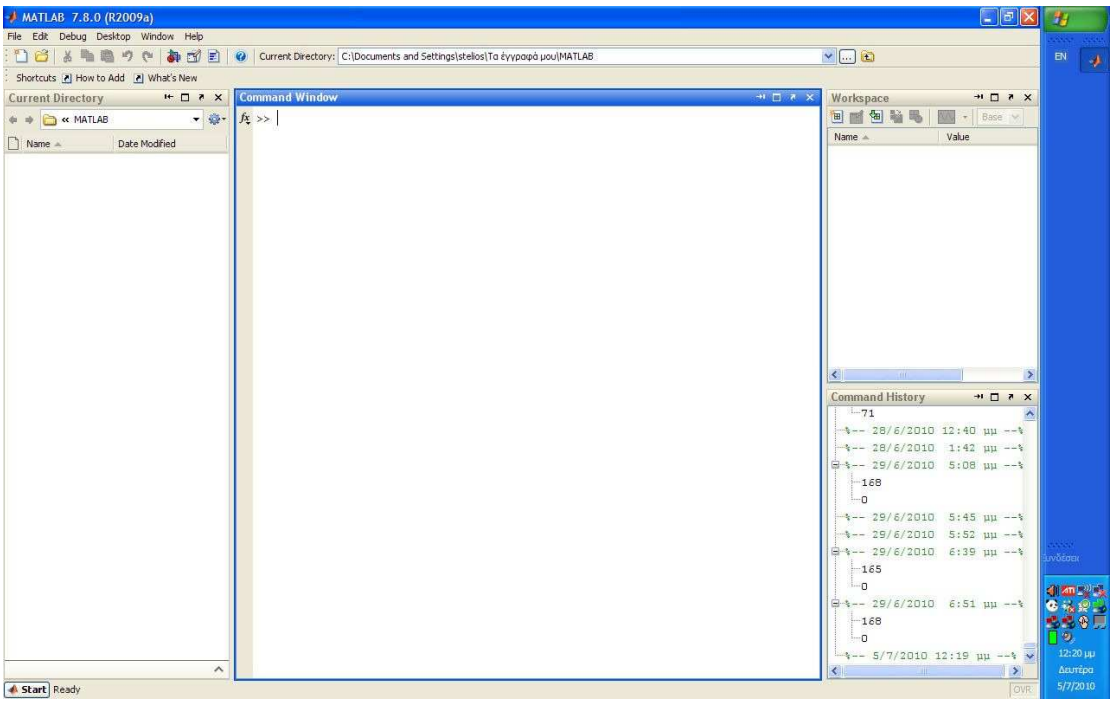
Έγινε κατανοητό από τα παραπάνω ότι τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB είναι προγραμματιστικά εργαλεία πολύ φιλικά για τον χρήστη. Φυσικά δεν παρουσιάσθηκαν σχεδόν καθόλου οι μεγάλες τους δυνατότητες, αλλά μπορεί κανείς να πει κατ'αρχήν ότι σε επίπεδο εντολών μπορούν να υλοποιήσουν κώδικες με τρόπο τουλάχιστον ανάλογο με τον τρόπο που γίνεται αυτό στη FORTRAN. Οι δυνατότητές τους επομένως είναι κατάλληλες για την υλοποίηση και κωδίκων για τον υπολογισμό θερμοφυσικών ιδιοτήτων εργαζόμενων μέσων, αρκεί να δίνονται οι κατάλληλες καταστατικές εξισώσεις και οι εξισώσεις ιδιοτήτων μεταφοράς. Μάλιστα, μία τέτοια υλοποίηση έχει γίνει για το ελαφρύ νερό σε περιβάλλον MATLAB και είναι διαθέσιμη εμπορικά (βλ. και Kelvin, 2001). Στο επόμενο $3^{ο}$ Κεφάλαιο, δίνονται τα βασικά στοιχεία αυτών των καταστατικών εξισώσεων και των εξισώσεων ιδιοτήτων για δύο από τα σημαντικότερα ψυκτικά μέσα στην τεχνολογία των Πυρηνικών Αντιδραστήρων Ισχύος, το ελαφρύ και το βαρύ νερό.

ΔΙΑΓΡΑΜΜΑΤΑ ΤΟΥ 2^{ου} ΚΕΦΑΛΑΙΟΥ



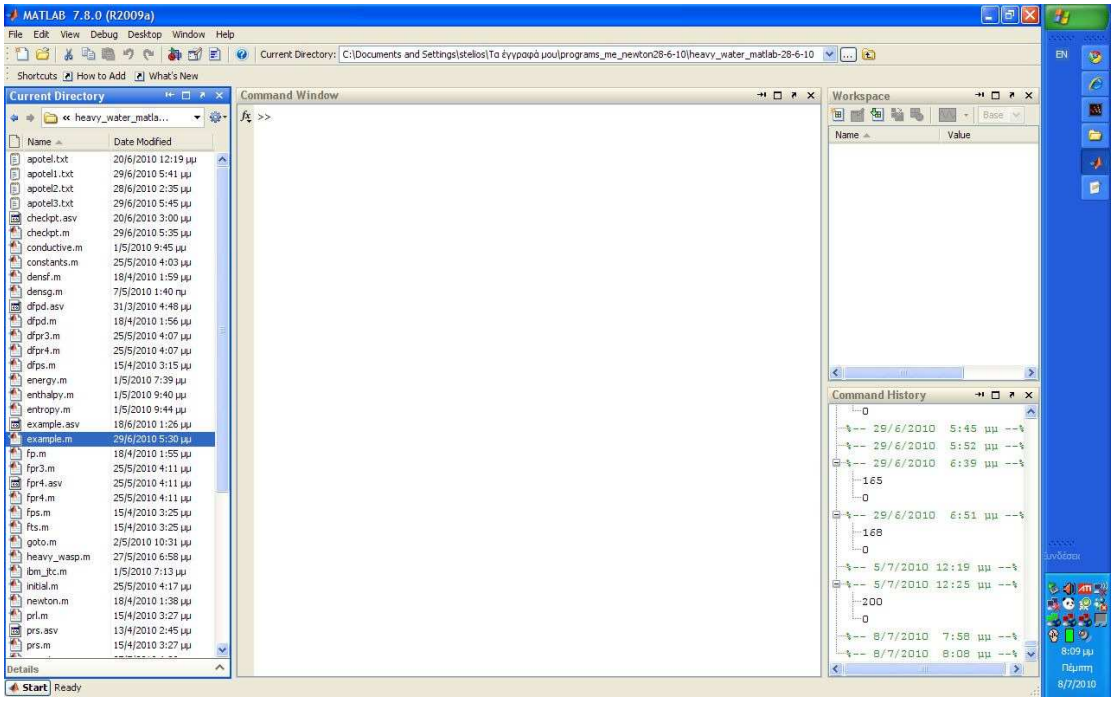
Διάγραμμα 2.1

MATLAB: Εκκίνηση (start/ Programs)



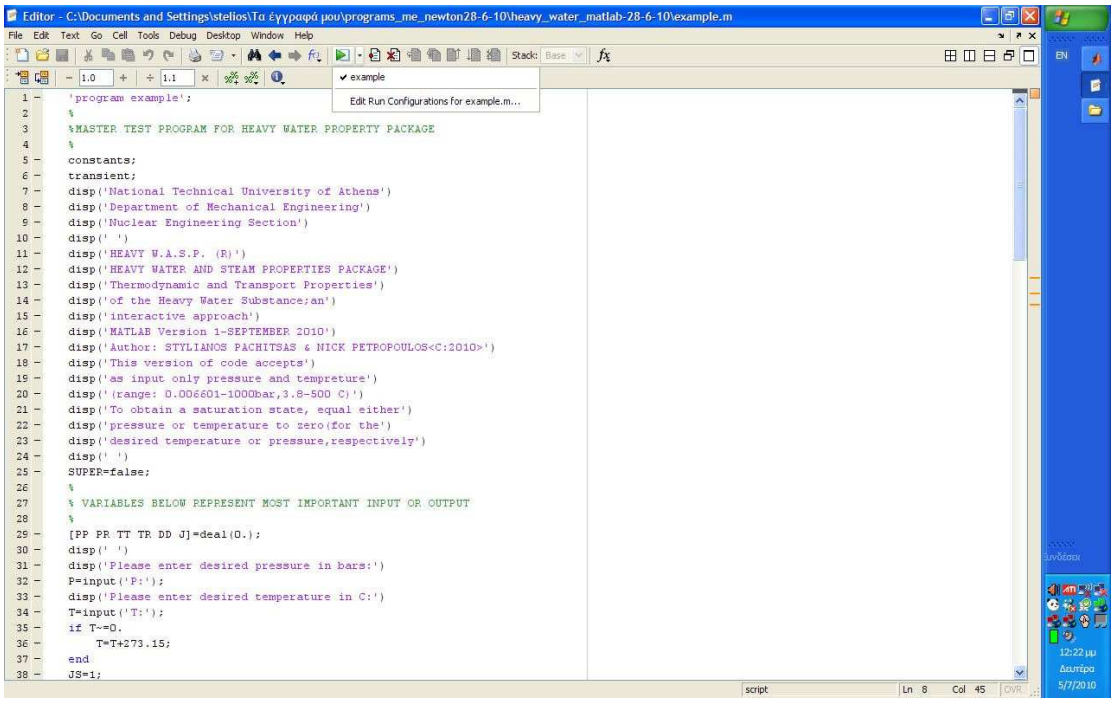
Διάγραμμα 2.2

MATLAB: Αρχικό περιβάλλον



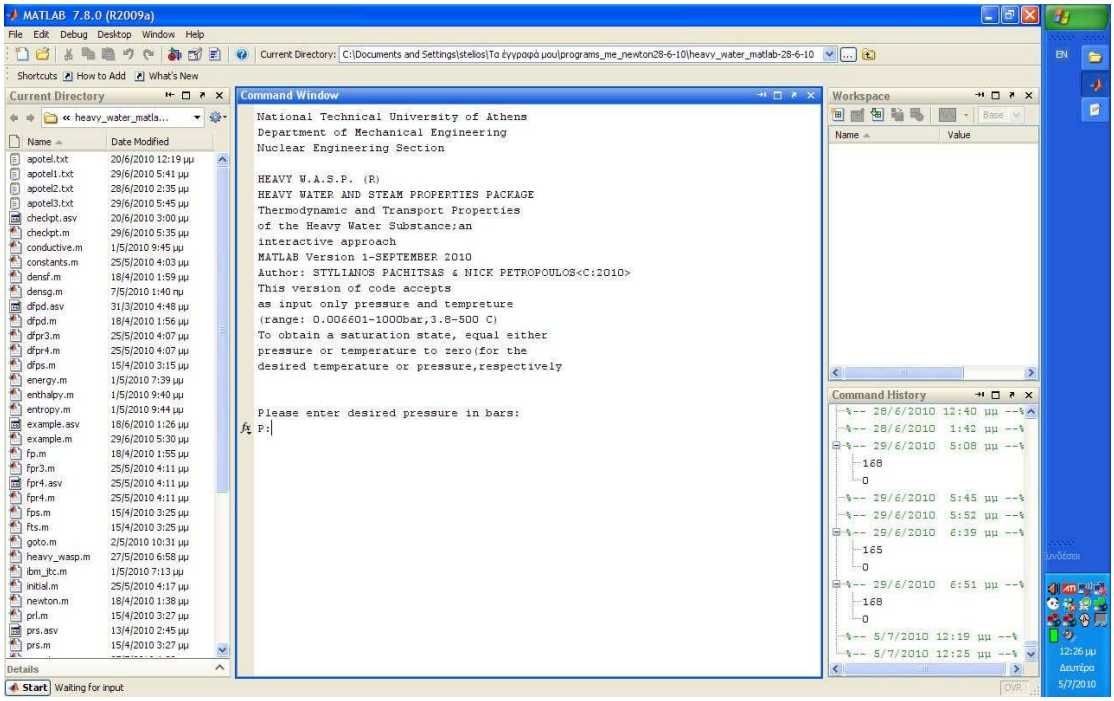
Διάγραμμα 2.3

MATLAB: Επιλογή προγράμματος .m από συγκεκριμένο φάκελο



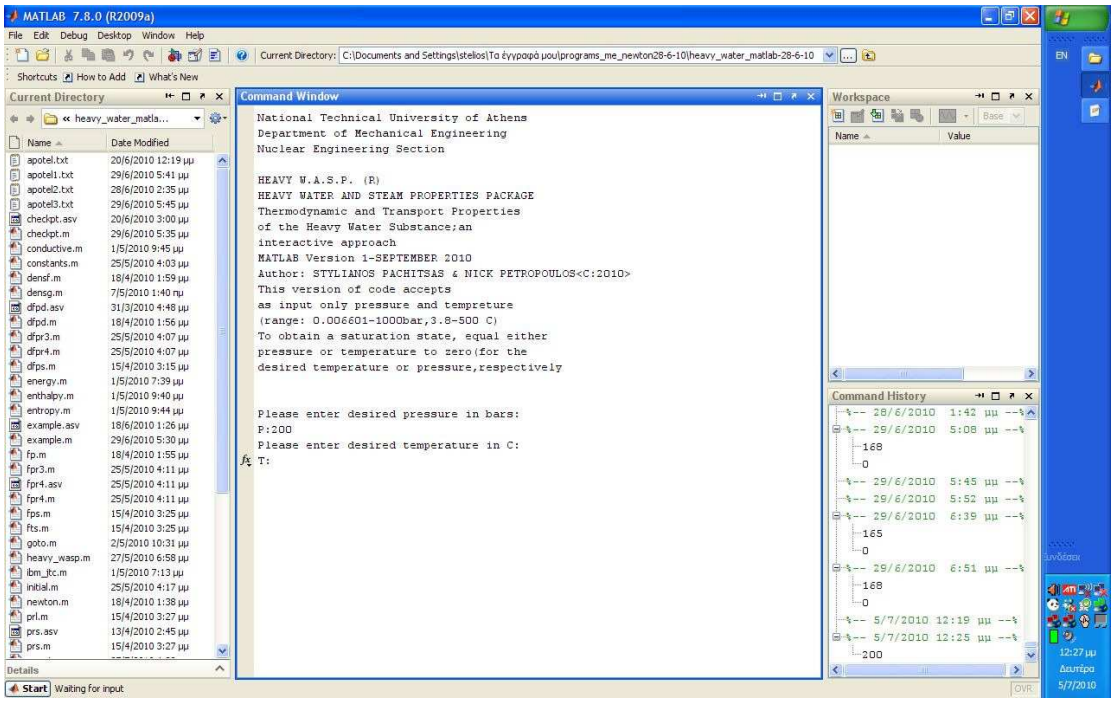
Διάγραμμα 2.4

MATLAB: Desktop Editor



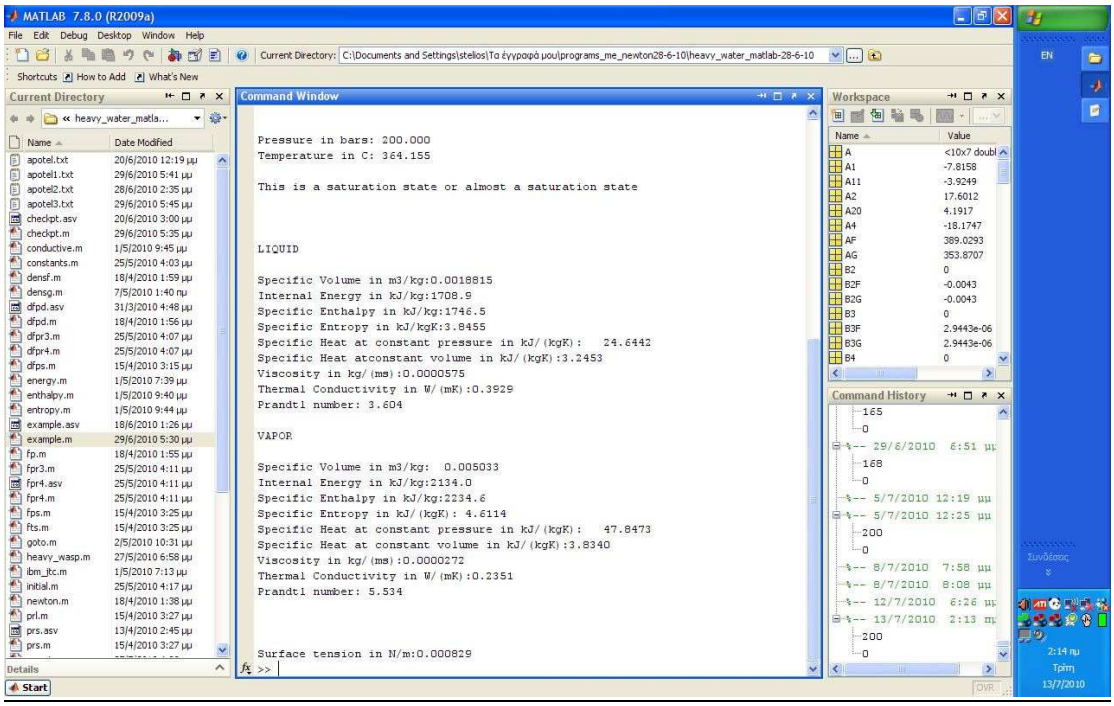
Διάγραμμα 2.5

MATLAB: Command Window



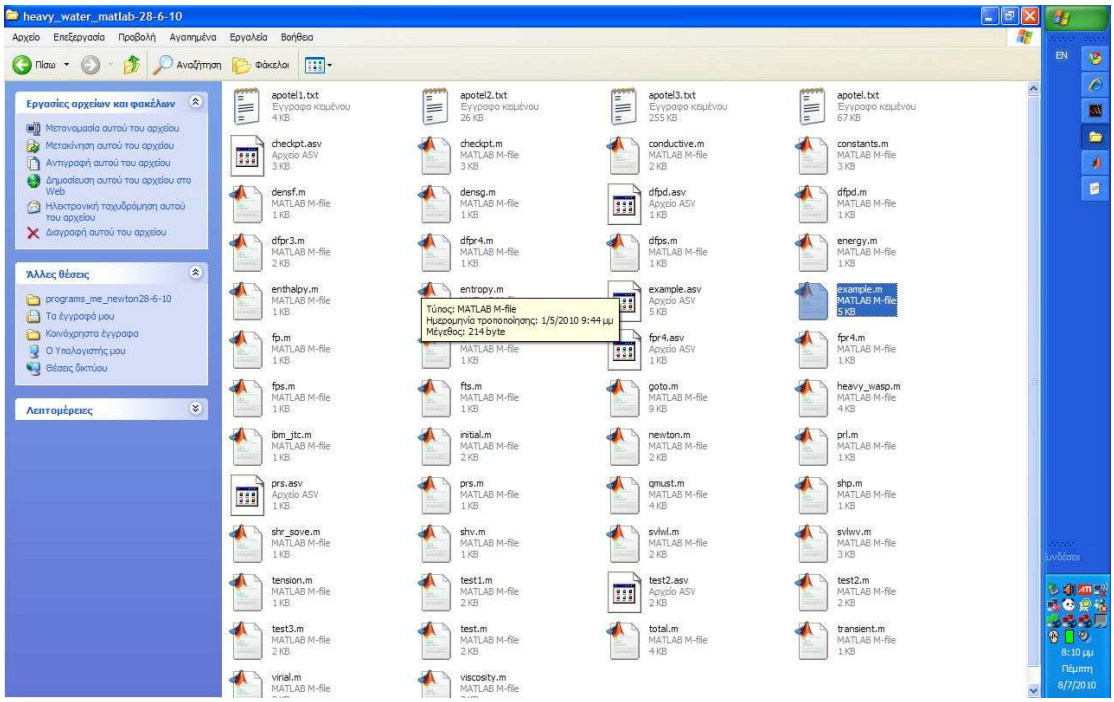
Διάγραμμα 2.6

MATLAB: Command Window, Εισαγωγή πίεσης και θερμοκρασίας



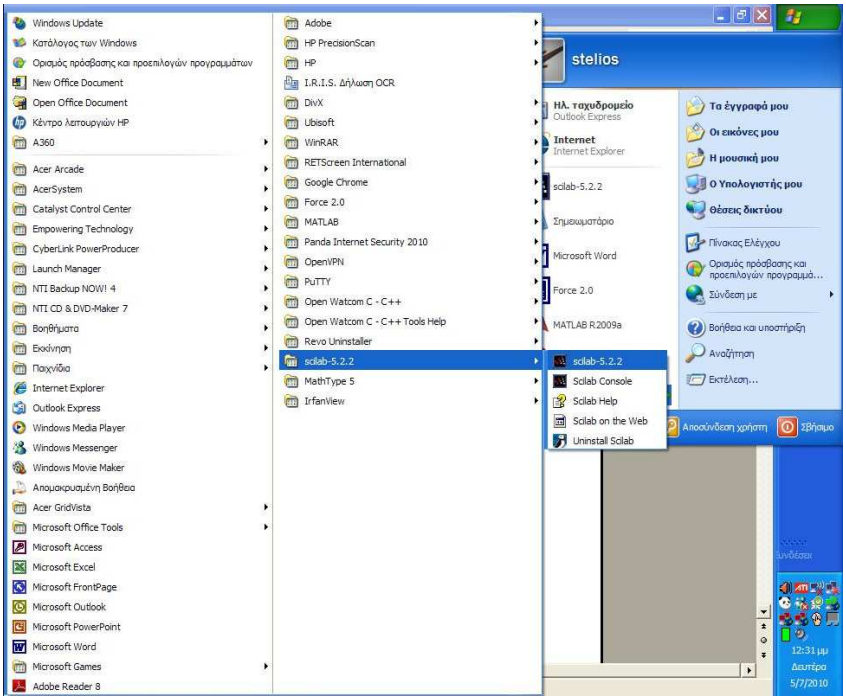
Διάγραμμα 2.7

MATLAB: Command Window, Αποτελέσματα



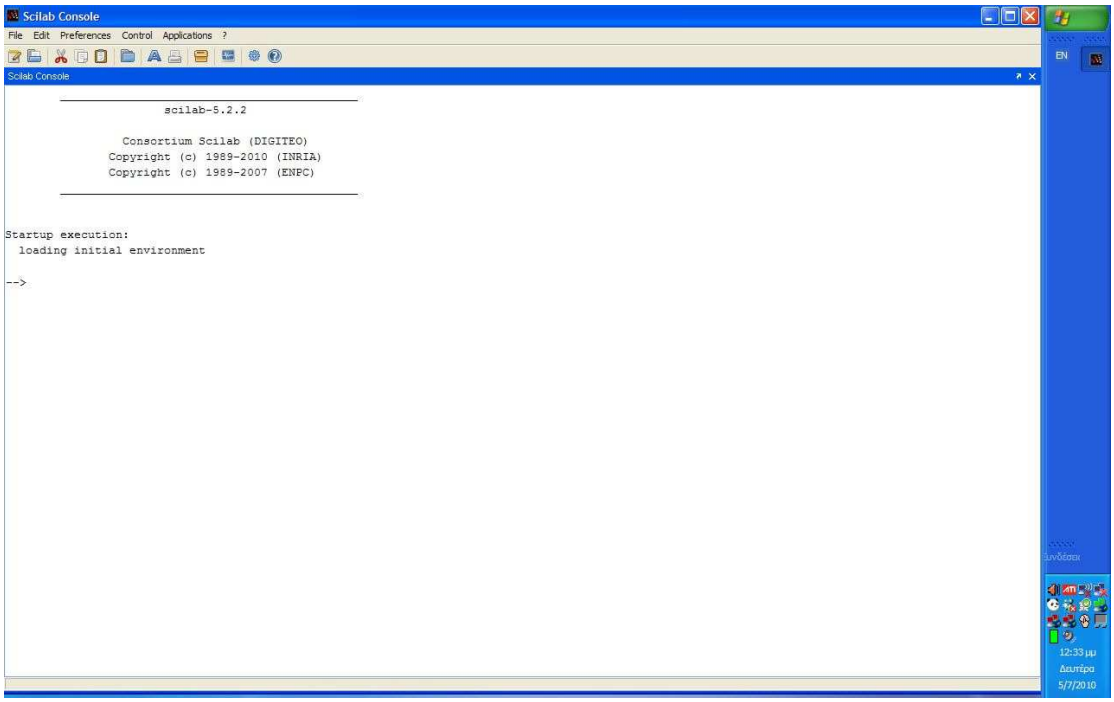
Διάγραμμα 2.8

MATLAB: Επιλογή εκτέλεσης προγράμματος με το "ποντίκι"



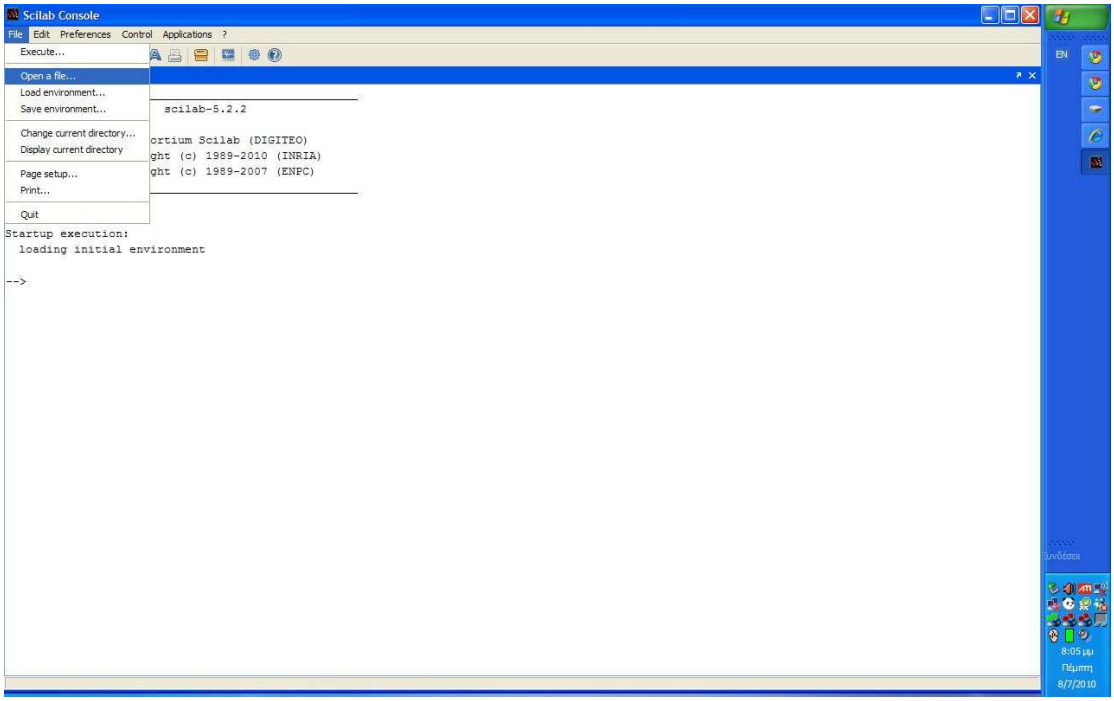
Διάγραμμα 2.9

SCILAB: Εκκίνηση (start/ Programs)



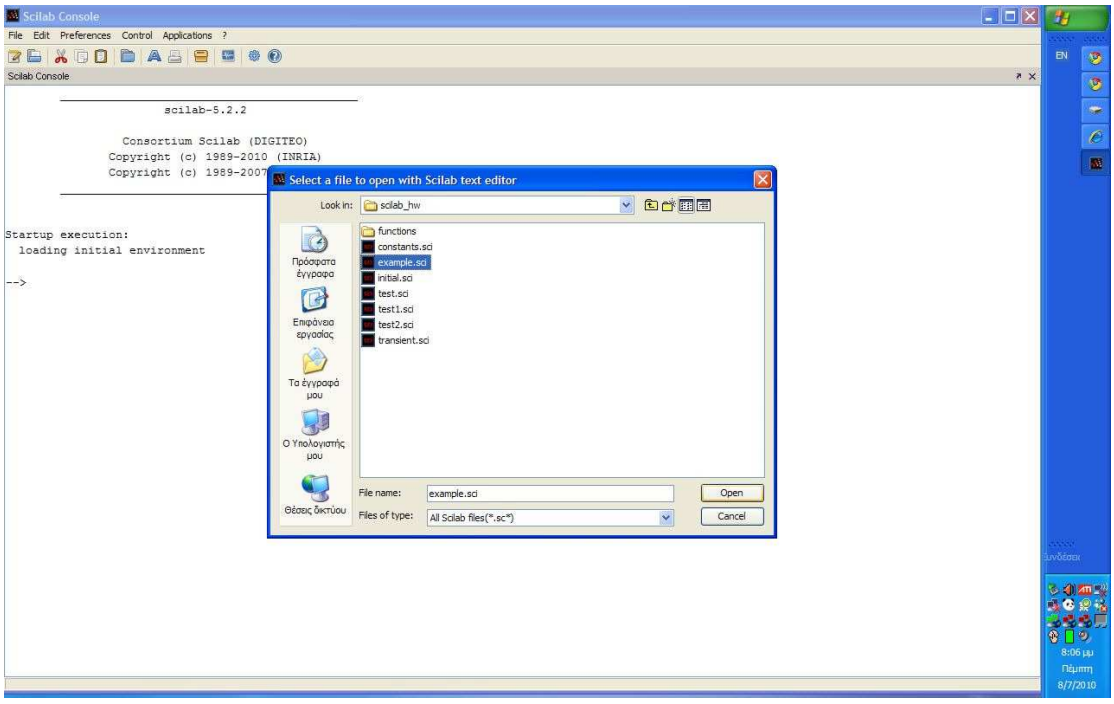
Διάγραμμα 2.10

SCILAB: Αρχικό περιβάλλον



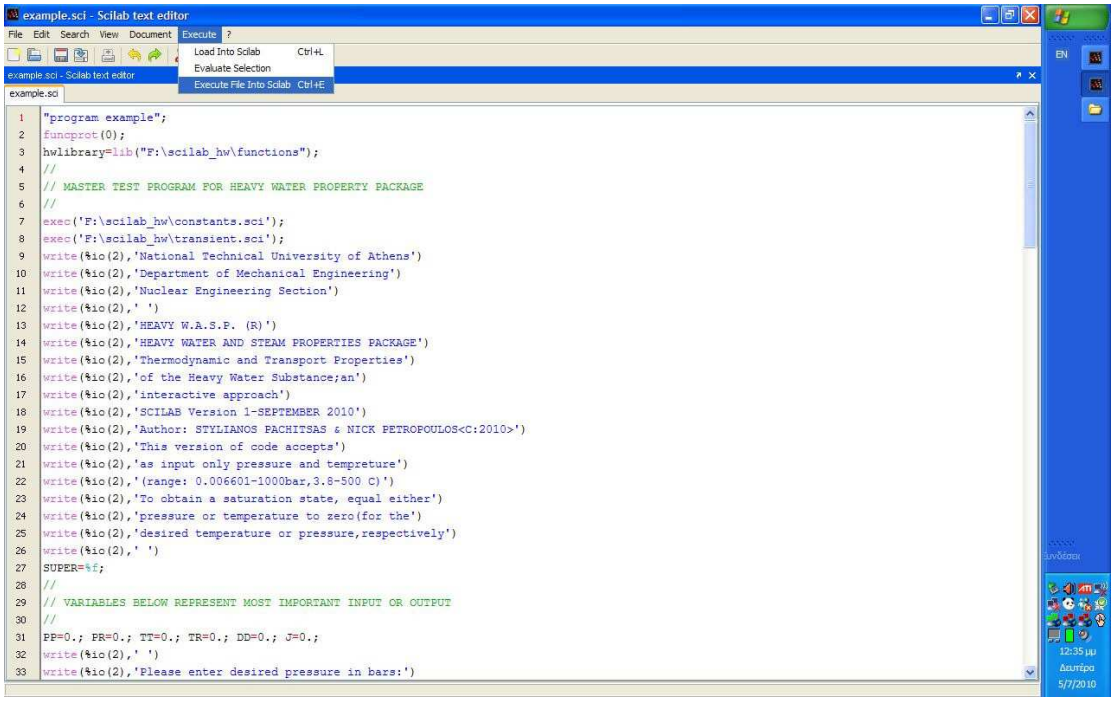
Διάγραμμα 2.11

SCILAB: Διαδικασία επιλογής προγράμματος: open



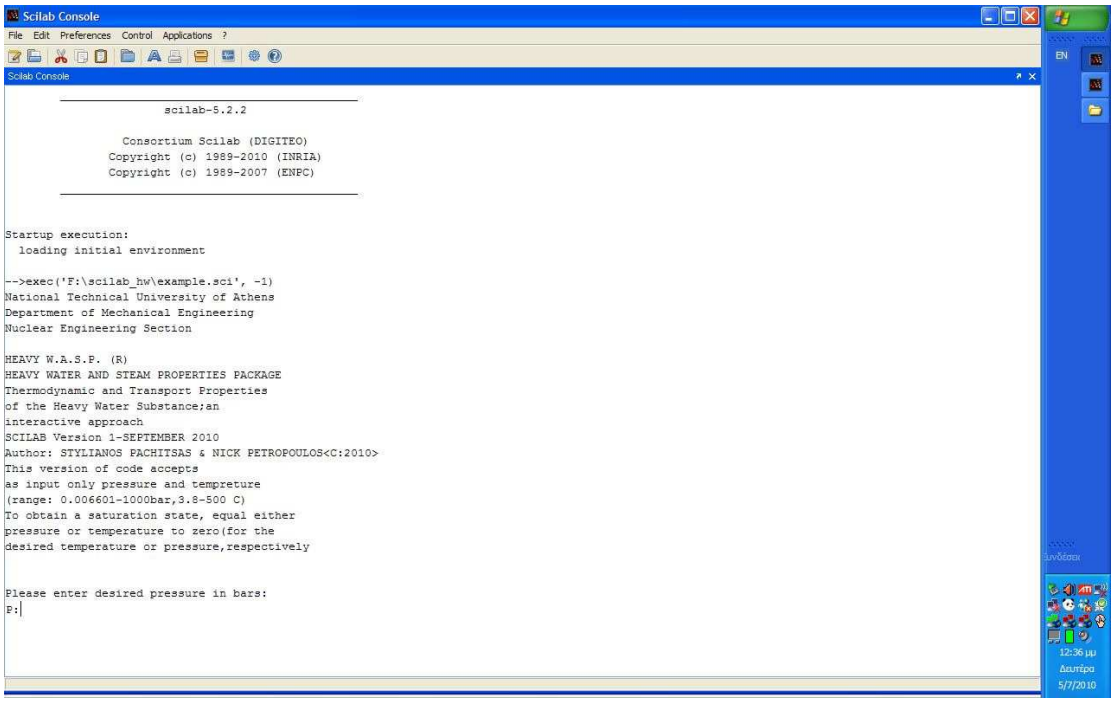
Διάγραμμα 2.12

SCILAB: Διαδικασία επιλογής προγράμματος: εύρεση φακέλου και αρχείου .sci



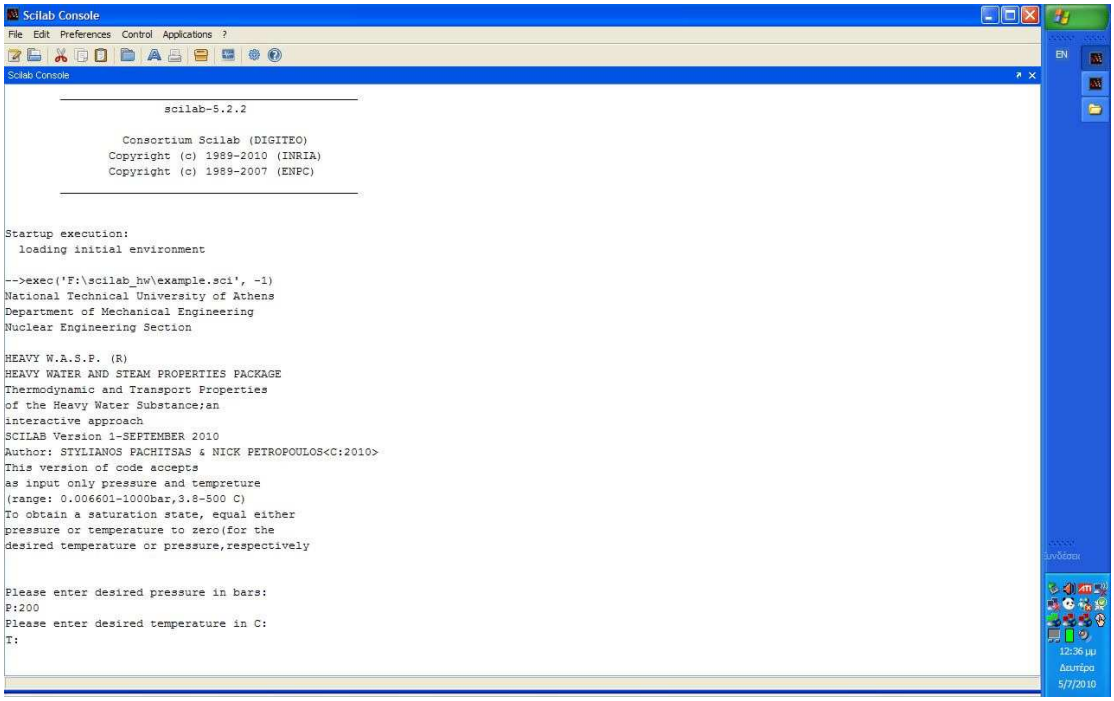
Διάγραμμα 2.13

SCILAB: Desktop Editor



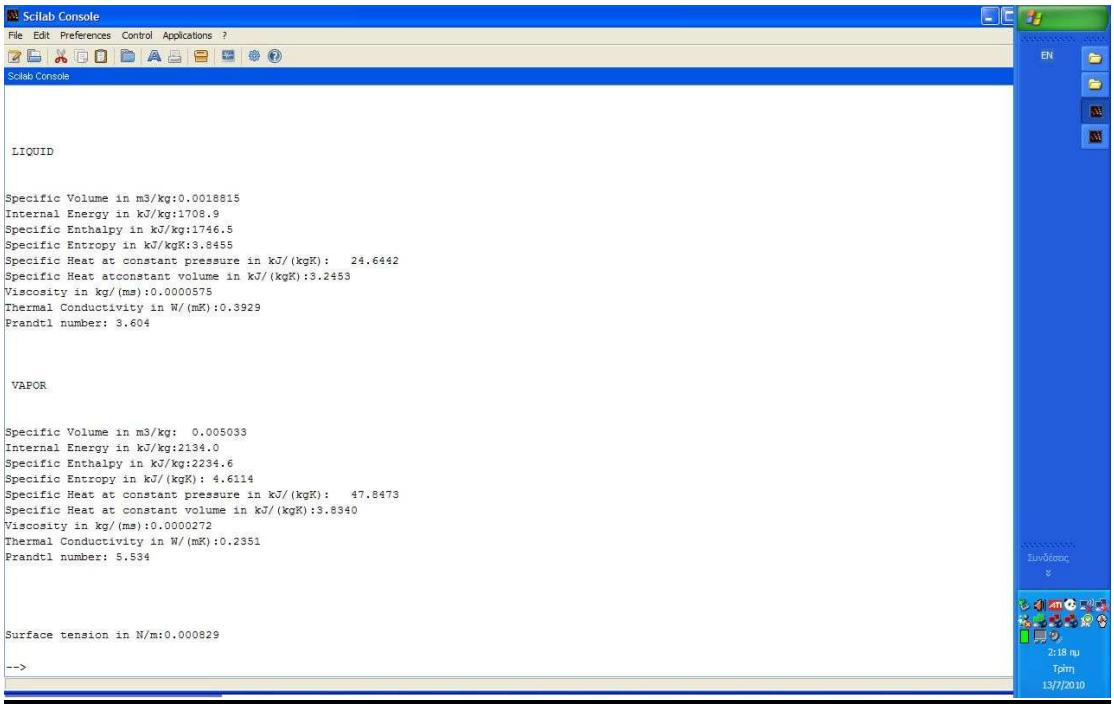
Διάγραμμα 2.14

SCILAB: Command Window



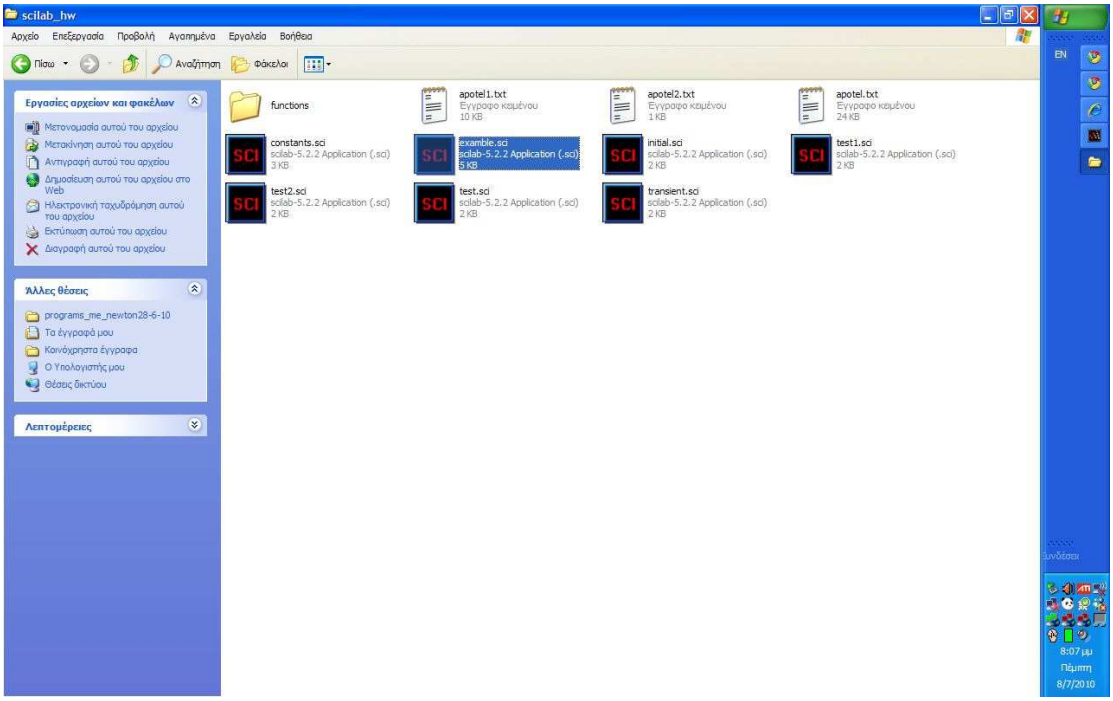
Διάγραμμα 2.15

SCILAB: Command Window, Εισαγωγή πίεσης και θερμοκρασίας



Διάγραμμα 2.16

SCILAB: Command Window, Αποτελέσματα



Διάγραμμα 2.17

SCILAB: Επιλογή εκτέλεσης προγράμματος με το "ποντίκι"

ΚΕΦΑΛΑΙΟ 3
ΘΕΡΜΟΔΥΝΑΜΙΚΑ ΣΥΝΕΠΕΙΣ ΚΑΤΑΣΤΑΤΙΚΕΣ ΕΞΙΣΩΣΕΙΣ ΚΑΙ ΕΞΙΣΩΣΕΙΣ ΙΔΙΟΤΗΤΩΝ ΜΕΤΑΦΟΡΑΣ ΓΙΑ ΤΟ ΕΛΑΦΡΥ ΚΑΙ ΤΟ ΒΑΡΥ ΥΔΩΡ

3.1 Εισαγωγή

Το ελαφρύ νερό είναι αδρανές, φθηνό και άμεσα διαθέσιμο. Χρησιμοποιείται κυρίως ως εργαζόμενο μέσο για μεταφορά ενθαλπίας σε ποικιλία μονάδων παραγωγής ενέργειας και εκμετάλλευσης. Τυποποιημένοι πίνακες θερμοφυσικών ιδιοτήτων (δηλ. θερμοδυναμικών και ιδιοτήτων μεταφοράς) ελαφρού νερού και ατμού είναι διαθέσιμοι στους μηχανικούς από το πρώτο μισό του περασμένου αιώνα. Το βαρύ νερό έχει παρόμοιες θερμοφυσικές ιδιότητες με το ελαφρύ παρόλαυτά δεν είναι φθηνό ούτε άμεσα διαθέσιμο. Χρησιμοποιείται κυρίως ως εργαζόμενο μέσο για μεταφορά ενθαλπίας σε πυρηνικούς αντιδραστήρες ισχύος τύπου CANDU. Τυποποιημένοι πίνακες θερμοφυσικών ιδιοτήτων βαρέος ύδατος και ατμού έγιναν διαθέσιμοι στους μηχανικούς περί το τέλος του περασμένου αιώνα. Τόσο για το ελαφρύ όσο και για το βαρύ νερό διατίθενται υπολογιστικοί κώδικες που συνήθως παρεμβάλλουν σε αυτούς τους πίνακες χρησιμοποιώντας μία ποικιλία καμπυλών και μεθόδων παρεμβολής. Πολλοί από αυτούς τους κώδικες είναι αργοί και παρουσιάζουν έλλειψη αξιοπιστίας αποτελεσμάτων ενώ κάποιοι άλλοι έχουν σχεδιαστεί για εξειδικευμένες χρήσεις και δεν παρέχουν πολλές από τις θερμοφυσικές ιδιότητες που ενδιαφέρουν. Στο παρόν Κεφάλαιο της Διπλωματικής Εργασίας παρουσιάζονται σύγχρονες καταστατικές εξισώσεις και άλλες εξισώσεις για το ελαφρύ και το βαρύ νερό για τον υπολογισμό των θερμοφυσικών ιδιοτήτων τους, οι οποίες μπορούν να χρησιμοποιηθούν σε κώδικες υπολογισμών.

3.2 Θερμοδυναμικά συνεπείς καταστατικές εξισώσεις

Οι θερμοδυναμικές ιδιότητες του ελαφρού και του βαρέος ύδατος μπορούν να υπολογίζονται αξιοποιώντας διατυπώσεις με βάση την εξίσωση ελεύθερης ενέργειας του Helmholtz, όπως παρουσιάζονται από τον Wagner (1998) για το ελαφρύ νερό, και από τον Hill (1981) για το βαρύ νερό. Άλλες εξισώσεις διατίθενται για τις ιδιότητες μεταφοράς για τα δύο αυτά εργαζόμενα μέσα. Οι διατυπώσεις αυτές για τις θερμοδυναμικές ιδιότητες χαρακτηρίζονται ως καταστατικές εξισώσεις

"θερμοδυναμικά συνεπείς" και το κύριο χαρακτηριστικό τους είναι ότι όλες οι θερμοδυναμικές ιδιότητες του εργαζόμενου μέσου προκύπτουν από απλές μαθηματικές πράξεις.

Μία καταστατική εξίσωση ειδικής ελεύθερης ενέργειας Helmholtz για τον υπολογισμό των θερμοφυσικών ιδιοτήτων μέσων όπως τα νερά, έχει τη μορφή:

$$\psi = \psi(\rho, \mathrm{T}) = \psi_o(\mathrm{T}) + RT[\ln \rho + \rho Q(\rho, \mathrm{T})] \qquad (3\text{-}1)$$

όπου

ψ είναι η ειδική ελεύθερη ενέργεια Helmholtz
$\psi_o(T)$ είναι πολυωνυμική συνάρτηση της θερμοκρασίας, και
$Q(\rho,T)$ είναι συνάρτηση που προέκυψε από προσαρμογή στις πειραματικές τιμές

και ικανοποιεί

- επαρκώς (αλλά όχι ακριβώς) τον περιορισμό ότι η ελεύθερη ενέργεια Gibbs για κορεσμένο νερό και ατμό πρέπει να είναι ίδια
- ότι στο κρίσιμο σημείο πρέπει

$$\left(\frac{\partial P}{\partial \rho}\right)_{\mathrm{T}} = 0$$

$$\left(\frac{\partial^2 P}{\partial \rho^2}\right)_{\mathrm{T}} = 0$$

και $\rho=\rho_c$, $T=T_c$, με τον δείκτη c συμβολίζεται το κρίσιμο σημείο.

Η καταστατική εξίσωση αυτή μπορεί να τροποποιηθεί και να πάρει εύκολα την περισσότερο αξιοποιήσιμη μορφή:

$$P = P(\rho, T) = \rho^2 \left(\frac{\partial \psi}{\partial \rho}\right)_{\mathrm{T}} \qquad (3\text{-}2)$$

όπου το καταστατικό μέγεθος της πίεσης P (MPa) εκφράζεται ως συνάρτηση των καταστατικών μεγεθών της πυκνότητας ρ (gcm^{-3}) και της θερμοκρασίας T (K).

Τα καταστατικά μεγέθη πίεσης, θερμοκρασίας και πυκνότητας του κρίσιμου σημείου που ικανοποιούν τις καταστατικές εξισώσεις (3-1) και (3-2) είναι δυνατόν να είναι διαφορετικά από τα πραγματικά κρίσιμα μεγέθη και ονομάζονται ψευδοκρίσιμα καταστατικά μεγέθη.

Ο προσδιορισμός των θερμοδυναμικών ιδιοτήτων των νερών από μία καταστατική εξίσωση όπως η (3-1) ή η (3-2) δηλ. της ενθαλπίας, της εντροπίας και των θερμικών θερμοχωρητικοτήτων μπορεί να γίνεται με απλές παραγωγίσεις, ως εξής:

ειδική εσωτερική ενέργεια $u = \left(\frac{\partial(\psi\tau)}{\partial\tau} \right)_\rho$ (kJkg^{-1}), $\tau = \frac{1000}{T}$

ενθαλπία $h = u + \frac{P}{\rho}$ (kJkg^{-1})

ειδική εντροπία $s = -\left(\frac{\partial\psi}{\partial T} \right)_\rho$ (kJkg^{-1}K^{-1})

ειδική θερμοχωρητικότητα υπό σταθερό όγκο $c_V = \left(\frac{\partial u}{\partial T} \right)_\rho$ (kJkg^{-1}K^{-1})

ειδική θερμοχωρητικότητα υπό σταθερή πίεση

$$c_P = \left(\frac{\partial h}{\partial T} \right)_\rho - \left(\frac{\partial h}{\partial \rho} \right)_T \left[\frac{\left(\frac{\partial P}{\partial T} \right)_\rho}{\left(\frac{\partial P}{\partial \rho} \right)_T} \right] \text{(kJkg}^{-1}\text{K}^{-1}\text{)}$$

συντελεστής ισεντροπικής μεταβολής $\gamma = \frac{C_P}{C_V}$

συντελεστής Joule-Thomson $\mu = \left(\frac{\partial T}{\partial P} \right)_h$ (KMPa$^{-1)}$

ταχύτητα του ήχου $a = 10^{3/2} \gamma \sqrt{\left(\frac{\partial P}{\partial \rho} \right)_T}$ (ms^{-1})

ισοθερμοκρασιακή συμπιεστότητα $k_T = \frac{1}{\rho \left(\frac{\partial P}{\partial \rho} \right)_T}$ (MPa^{-1})

Στην προγραμματιστική υλοποίηση της παρούσας Διπλωματικής Εργασίας για μεν το ελαφρύ νερό χρησιμοποιείται η θεμελιώδης καταστατική εξίσωση του Keyes (1968), και για το βαρύ νερό η θεμελιώδης καταστατική εξίσωση του Hill (1981). Η καταστατική εξίσωση του Keyes (1968) δεν είναι η πλέον σύγχρονη, όπως αυτή που παρουσιάζεται στον Wagner (1998), παρόλαυτά υπολογίζει τις θερμοδυναμικές ιδιότητες του ελαφρού ύδατος με ικανοποιητική ακρίβεια για βιομηχανική χρήση. Η καταστατική εξίσωση του Hill (1981) είναι η τελευταία που παρουσιάσθηκε για το

βαρύ ύδωρ και υπολογίζει τις θερμοδυναμικές του ιδιότητες με ικανοποιητική ακρίβεια επίσης για βιομηχανική χρήση.

3.3 Εξισώσεις ιδιοτήτων μεταφοράς

για το ελαφρύ ύδωρ

Η συνεκτικότητα η ($kgm^{-1}s^{-1}$) και η θερμική αγωγιμότητα λ ($Wm^{-1}K^{-1}$) του ελαφρού ύδατος μπορεί να υπολογίζονται ως συναρτήσεις της πυκνότητας ρ και της θερμοκρασίας Τ από τις σχέσεις που δίνει ο Sengers (1986). Η επιφανειακή τάση σ (Nm^{-2}) του ελαφρού ύδατος μπορεί να υπολογίζεται ως συνάρτηση της θερμοκρασίας Τ από τη σχέση που δίνει ο Straub (1980).

για το βαρύ ύδωρ

Η συνεκτικότητα η ($kgm^{-1}s^{-1}$) και η θερμική αγωγιμότητα λ ($Wm^{-1}K^{-1}$) του βαρέος ύδατος μπορεί να υπολογίζονται ως συναρτήσεις της πυκνότητας ρ και της θερμοκρασίας Τ από τις σχέσεις που δίνει ο Matsunaga (1983). Η επιφανειακή τάση σ (Nm^{-2}) του βαρέος ύδατος μπορεί να υπολογίζεται ως συνάρτηση της θερμοκρασίας Τ από τη σχέση που δίνει ο Straub (1980).

3.4 Η περιοχή ισχύος των εξισώσεων

για το ελαφρύ ύδωρ

Σε ό,τι αφορά στις θερμοφυσικές ιδιότητες του ελαφρού ύδατος, αναφέρθηκε ήδη ότι οι καταλληλότερες σχέσεις και συσχετίσεις που επελέγησαν να χρησιμοποιηθούν στο αναπτύχθηκαν από τους:

Keyes (1968)
(καταστατική εξίσωση και θερμοδυναμικές ιδιότητες)
Sengers (1986)
(δυναμική συνεκτικότητα και θερμική αγωγιμότητα)
Straub (1980)
(επιφανειακή τάση)

Στην Διδακτορική Διατριβή του Πετρόπουλου (2003), παρουσιάσθηκαν επίσης οι αντίστοιχες περιοχές πιέσεων και θερμοκρασιών για τις οποίες ισχύουν οι χρησιμοποιούμενες συσχετίσεις για τις θερμοφυσικές ιδιότητες τον ελαφρού ύδατος.

Από τη μελέτη των σχετικών δεδομένων προκύπτει ότι:

Η ελάχιστη κοινή περιοχή ισχύος των χρησιμοποιούμενων συσχετίσεων σχετικά με τις θερμοφυσικές ιδιότητες του ελαφρού ύδατος μπορεί να ορισθεί ως:

Πιέσεις: 0 - 1000 bar

Θερμοκρασίες: 0 -1500 ^{o}C

Εξαιρείται μικρή περιοχή στη γειτονιά του κρίσιμου σημείου, ως εξής:

$|T-T^*| \leq 10K$

$|\rho/\rho^*-1| \leq 0.3$

Το κρίσιμο σημείο ορίζεται για τις συσχετίσεις των ιδιοτήτων, με τις ψευδοκρίσιμες τιμές της πίεσης, της θερμοκρασίας και της πυκνότητας, ως εξής:

ψευδοκρίσιμη πίεση $P^* = 221.15$ bar

ψευδοκρίσιμη θερμοκρασία $T^* = 647.27$ K και

ψευδοκρίσιμη πυκνότητα $\rho^* = 317.763$ kgm^{-3}

Λόγω επιφυλάξεων που εκφράζονται για την ακρίβεια της συσχετίσεως για την επιφανειακή τάση του ελαφρού ύδατος, που προτείνεται από τον Straub(1980) χρησιμοποιείται η τιμή για τη θερμοκρασία στο κρίσιμο σημείο που δίνουν οι ίδιοι οι δημιουργοί της συσχέτισης. Δηλαδή:

$T^* = 647.15$ K

Η κοινή περιοχή ισχύος των χρησιμοποιούμενων εξισώσεων για τις θερμοφυσικές ιδιότητες εξυπηρετεί σε θερμοδυναμικούς υπολογισμούς για την περιοχή πιέσεων και θερμοκρασιών στην οποία εργάζονται υπό κανονικές συνθήκες οι Πυρηνικοί Αντιδραστήρες Ισχύος

<u>για το βαρύ ύδωρ</u>

Σε ό,τι αφορά στις θερμοφυσικές ιδιότητες του βαρέος ύδατος, αναφέρθηκε ήδη ότι οι καταλληλότερες σχέσεις και συσχετίσεις που επελέγησαν να χρησιμοποιηθούν στο αναπτύχθηκαν από τους:

Hill (1981)

(καταστατική εξίσωση και θερμοδυναμικές ιδιότητες)

Matsunaga (1983)

(δυναμική συνεκτικότητα και θερμική αγωγιμότητα)

Straub (1980)

(επιφανειακή τάση)

Στην Διδακτορική Διατριβή του Πετρόπουλου (2003), παρουσιάσθηκαν επίσης οι αντίστοιχες περιοχές πιέσεων και θερμοκρασιών για τις οποίες ισχύουν οι χρησιμοποιούμενες συσχετίσεις για τις θερμοφυσικές ιδιότητες του βαρέος ύδατος.

Από τη μελέτη των σχετικών δεδομένων προκύπτει ότι:

Η ελάχιστη κοινή περιοχή ισχύος των χρησιμοποιούμενων συσχετίσεων σχετικά με τις θερμοφυσικές ιδιότητες του βαρέος ύδατος μπορεί να ορισθεί ως:
Πιέσεις: 0.006601 - 1000 bar
Θερμοκρασίες: 3.8 -600 ^{o}C
Εξαιρείται μικρή περιοχή στη γειτονιά του κρίσιμου σημείου, ως εξής:

$|T - T^*| \leq 10 K$

$|\rho/\rho^* - 1| \leq 0.3$

Το κρίσιμο σημείο ορίζεται για τις συσχετίσεις των ιδιοτήτων, με τις ψευδοκρίσιμες τιμές της πίεσης, της θερμοκρασίας και της πυκνότητας, ως εξής:

ψευδοκρίσιμη πίεση $P^* = 216.6$ bar
ψευδοκρίσιμη Θερμοκρασία $T^* = 643.89$ K και
ψευδοκρίσιμη πυκνότητα $\rho^* = 358$ kgm^{-3}

Λόγω επιφυλάξεων που εκφράζονται για την ακρίβεια της συσχετίσεως για την επιφανειακή τάση του ελαφρού ύδατος, που προτείνεται από τον Straub(1980) χρησιμοποιείται η τιμή για τη θερμοκρασία στο κρίσιμο σημείο που δίνουν οι ίδιοι οι δημιουργοί της συσχέτισης. Δηλαδή:

$T^* = 644.65$ K

Η κοινή περιοχή ισχύος των χρησιμοποιούμενων εξισώσεων για τις θερμοφυσικές ιδιότητες εξυπηρετεί σε θερμοδυναμικούς υπολογισμούς για την περιοχή πιέσεων και θερμοκρασιών στην οποία εργάζονται υπό κανονικές συνθήκες οι Πυρηνικοί Αντιδραστήρες Ισχύος.

3.4 Πίεση κορεσμού

Η πίεση κορεσμού Ps δίνεται ως συνάρτηση της θερμοκρασίας κορεσμού από κατάλληλες συσχετίσεις, τόσο για το ελαφρύ όσο και για το βαρύ ύδωρ.

Οι συσχετίσεις αυτές πρέπει να ικανοποιούν την ισότητα:

$$(\psi + P/\rho)\text{κορεσμένου υγρού} = (\psi + P/\rho)\text{κορεσμένου ατμού}$$

για το ελαφρύ ύδωρ

Η συσχέτιση έχει διατυπωθεί στον Schmidt (1987) ως εξής:

$$P_s(T) = P^* \exp\left\{ \frac{\sum_{\nu=1}^{5} k_\nu (1-T_r)^\nu}{T_r + k_6(1-T_r)T_r + k_7(1-T_r)^2 T_r} - \frac{1-T_r}{k_8(1-T_r)^2 + k_9} \right\} \qquad (3\text{-}3)$$

με T_r = T/T* και P*, T* οι ψευδοκρίσιμες τιμές θερμοκρασίας και πίεσης του ελαφρού ύδατος.

Οι σταθερές k_i δίνονται και στον Πετρόπουλο (2003).

για το βαρύ ύδωρ

Η συσχέτιση έχει διατυπωθεί στον Hill (1982) ως εξής:

$$P_s(T) = P^* \exp\left[(T^*/T)\left(n_1\tau + n_2\tau^{1.9} + n_4\tau^2 + n_{11}\tau^{5.5} + n_{20}\tau^{10} \right) \right] \qquad (3\text{-}4)$$

με P*, T* οι ψευδοκρίσιμες τιμές θερμοκρασίας και πίεσης του βαρέος ύδατος και τ = 1000/T σε K^{-1}

Οι σταθερές n_i δίνονται και στον Πετρόπουλο (2003).

3.5 Υπολογισμός της πυκνότητας από τις καταστατικές εξισώσεις

Για τον υπολογισμό των θερμοδυναμικών ιδιοτήτων και των ιδιοτήτων μεταφοράς του ελαφρού ύδατος ή του βαρέος ύδατος από το σύνολο των σχέσεων που παρουσιάσθηκαν προηγούμενα, αρκεί να είναι γνωστή η πυκνότητα ρ και η θερμοκρασία Τ και να διατίθενται εξισώσεις για τις ιδιότητες μεταφοράς, οι οποίες δέχονται ως ανεξάρτητες μεταβλητές την πυκνότητα και τη θερμοκρασία. Το ζητούμενο όμως συνήθως είναι να μπορούν να υπολογισθούν οι θερμοφυσικές ιδιότητες του ελαφρού και του βαρέος ύδατος αν είναι γνωστά η πίεση Ρ και η θερμοκρασία Τ, πράγμα που ειδικά εξυπηρετεί σε κώδικες θερμοϋδραυλικής ανάλυσης Πυρηνικών Αντιδραστήρων Ισχύος. Με δεδομένες λοιπόν την πίεση και τη θερμοκρασία πρέπει με κάποιο τρόπο να προσδιορίζεται η αντίστοιχη πυκνότητα έτσι ώστε να μπορεί να διαμορφωθεί το απαραίτητο ζεύγος εισόδου πυκνότητας - θερμοκρασίας, το οποίο επιτρέπει τελικά τον εύκολο υπολογισμό όλων των ιδιοτήτων. Για τη διαδικασία αυτή διατίθεται η καταστατική εξίσωση (3-2) που είναι μία συνάρτηση της μορφής P = P(T,ρ), η οποία είναι πεπλεγμένη ως προς την πυκνότητα.

Αυτό αποκλείει κάθε συμβατική προσπάθεια για την αναλυτική της επίλυση ως προς ρ. Πρέπει να καταφύγει κανείς λοιπόν σε κάποια από τις γνωστές αριθμητικές μεθόδους επιλύσεως τέτοιων προβλημάτων. Η εφαρμογή μιας αριθμητικής μεθόδου προϋποθέτει επιπλέον την ύπαρξη καλής αρχικής προσέγγισης της τιμής της πυκνότητας που ζητείται να υπολογισθεί. Σύμφωνα με τον Hendricks (1973), οι αριθμητικές μέθοδοι που είναι κατάλληλες να χρησιμοποιηθούν για το σκοπό της αριθμητικής επίλυσης της καταστατικής εξίσωσης (3-2), της μορφής P = P(T,ρ) ως προς την πυκνότητα ρ, είναι η μέθοδος της διχοτομήσεως και η μέθοδος Newton-Raphson. Στον Πετρόπουλο (2003), προτιμάται η μέθοδος Newton-Raphson. Το κυριότερο σημείο που ενδιαφέρει μετά την επιλογή της αριθμητικής μεθόδου είναι ο τρόπος με τον οποίο θα εξασφαλισθεί η απαίτηση της μεθόδου για μία καλή αρχική προσέγγιση της ζητούμενης λύσης. Ενδιαφέρει λοιπόν να βρεθεί τρόπος να υπολογισθεί μία τιμή της πυκνότητας ρ του ελαφρού και του βαρέος ύδατος από τη δεδομένη πίεση P και θερμοκρασία T, η οποία να είναι αρκετά κοντά στην πραγματική, για να χρησιμοποιηθεί ως τέτοια προσέγγιση. Ο Πετρόπουλος (2003) προτείνει να χρησιμοποιηθεί μέθοδος που να ικανοποιεί τα ακόλουθα κριτήρια:

1. Να μην χρησιμοποιηθούν εξισώσεις και συσχετίσεις με μειωμένη αξιοπιστία και περιορισμένη περιοχή ισχύος.
2. Να υπάρχει δυνατότητα για την προσέγγιση της πυκνότητας τόσο στην υγρή, όσο και στην ατμώδη και την υπερκρίσιμη φάση του ελαφρού και του βαρέος ύδατος και σε ένα πεδίο θερμοκρασιών και πιέσεων τουλάχιστον ίσης έκτασης με την κοινή περιοχή ισχύος της καταστατικής εξίσωσης και των εξισώσεων για τις ιδιότητες μεταφοράς.
3. Η προσέγγιση της πυκνότητας να γίνεται με τη μέγιστη δυνατή ακρίβεια.
4. Η προσέγγιση να γίνεται με τη χρησιμοποίηση σχέσεων που παρουσιάζουν συνέχεια στις τιμές *στα* όρια μεταξύ των διαφόρων φάσεων και ειδικότερα στο όριο μεταξύ της υγρής και της υπερκρίσιμης και στο όριο μεταξύ της ατμώδους και της υπερκρίσιμης φάσης.

Σύμφωνα με αυτά τα κριτήρια η προσέγγιση για την πυκνότητα του ελαφρού και του βαρέος ύδατος στην υγρή και την ατμώδη φάση επιλέχθηκε να υπολογισθεί σύμφωνα με την καταστατική εξίσωση του Schmidt (1987) για το ελαφρύ ύδωρ (που είναι η διατύπωση για τις θερμοδυναμικές ιδιότητες του ελαφρού ύδατος IFC-1967).

Πρόκειται για σύνολο συσχετίσεων που συγκροτούν καταστατική εξίσωση υψηλής θερμοδυναμικής συνέπειας -συνάρτηση Gíbbs G = G(P,T), από τις οποίες συσχετίσεις παράγονται εύκολα συναρτήσεις που υπολογίζουν την πυκνότητα (ισοδύναμα τον ειδικό όγκο του ε.υ.).

Για τον υπολογισμό της αρχικής προσέγγισης της πυκνότητας σύμφωνα με την διατύπωση IFC-1967 για το βαρύ ύδωρ, χρειάζεται να γίνουν απλές τροποποιήσεις με βάση την αρχή των αντιστοιχουσών καταστάσεων (βλ. και Ulrych, 1981).

<u>για το ελαφρύ ύδωρ</u>

Τα κύρια σημεία της διατύπωσης IFC-1967 που ενδιαφέρουν έχουν ως εξής:

Η συνολική περιοχή ισχύος της τυποποίησης IFC-1967 καλύπτει την περιοχή πιέσεων $0 \leq P \leq 1000$ bar και την περιοχή θερμοκρασιών $0 \leq T \leq 1073.15$. Οι ψευδοκρίσιμες τιμές για τα καταστατικά μεγέθη πιέσεως, θερμοκρασίας και ειδικού όγκου στο κρίσιμο σημείο είναι: $P^* = 221.2$ bar, $T^* = 647.3$ K και $\nu^* = 0.00317 m^3 kg^{-1}$.

Η συνολική περιοχή ισχύος της διατύπωσης διαχωρίζεται σε έξι υποπεριοχές, οι οποίες απεικονίζονται στα Διαγράμματα 3.1 και 3.2 και καθορίζονται πιο κάτω:

Πίνακας 3.1

Υποπεριοχές ισχύος της διατύπωσης IFC – 1967

Περιοχή θερμοκρασίας	**Περιοχή Πίεσης**	**Υποπεριοχή**
	$0 \le P_r \le P_{rs}(Tr)$	**2**
$T_{rt} \le T_r \le T_{r1}$	$P_r = P_{rs}(T_r)$	**6**
	$P_{rs}(T_t) < P_r \le P_{r2}$	**1**

όπου $T_{rt} = Tt / T^* = 4.219990731 \times 10^{-1}$

με $Tt = 273.16K$ η θερμοκρασία στο τριπλό σημείο του ελαφρού ύδατος

	$0 \le P_r \le P_{rL}(T_r)$	**2**
$T_{r1} < T_r < T_{r2}$	$P_{rL}(T_r) < P_r < P_{rs}(T_r)$	**3**
	$P_r = P_{rs}(T_r)$	**5**
	$P_{rs}(T_r) < P_r \le P_{r2}$	**4**
$1 \le T_r \le T_{r2}$	$0 \le P_r \le P_{rL}(T_r)$	**2**
	$P_{rL}(T_r) < P_r \le P_{r2}$	**3**
$T_{r2} \le T_r \le T_{r3}$	$0 \le P_r \le P_{r2}$	**2**

Για την ευκολότερη κατανόηση τον διαστήματος των πιέσεων και των θερμοκρασιών που καλύπτουν οι υποπεριοχές που εμφανίζονται στον Πίνακα 3.1 και στα Διαγράμματα 3.1 και 3.2, διευκρινίζονται τα εξής:

- **Η υποπεριοχή 1** περιγράφει την υγρή φάση και ένα μέρος της υπερκρίσιμης φάσης μακριά από το κρίσιμο σημείο.
- **Η υποπεριοχή 2** περιγράφει την ατμώδη φάση και ένα μέρας της υπερκρίσιμης φάσης μακριά από το κρίσιμο σημείο.
- **Η υποπεριοχή 3** καλύπτει την ατμώδη φάση και ένα μέρας της υπερκρίσιμης φάσης γύρω και πάνω από το κρίσιμο σημείο.
- **Η υποπεριοχή 4** καλύπτει την υγρή φάση και ένα μέρος της υπερκρίσιμης φάσης για πιέσεις γύρω και πάνω από το κρίσιμο σημείο και για θερμοκρασίες στη γειτονιά του κρίσιμού σημείου.
- **Η υποπεριοχή 5** αντιπροσωπεύει καταστάσεις κορεσμού στη γειτονιά τον κρίσιμού σημείου.
- **Η υποπεριοχή 6** αντιπροσωπεύει καταστάσεις κορεσμού μακριά από το κρίσιμο σημείο.

Οι σταθερές T_{r1}, T_{r2}, T_{r3} και P_{r2} παίρνουν τις ακόλουθες τιμές:

$T_{r1}=9.626911787\times10^{-1}$

$T_{r2}=1.333462073$

$T_{r3}=T_{max}/T^{*}$, $T_{max}=12073.15K$, άρα $T_{r3}=1.657886606$

$P_{r2}=P_{max}/P^{*}$, $P_{max}=1000bar$, άρα $P_{r2}=4.616805171$

Ακόμα:

$P_{r1}=P_{rs}(T_{r1})$, $P_{rs}=P_{rs}(T_r)$ και $P_{rL}=P_{rL}(T_r)$ όπου $P_{rs}(T_r)$ και $P_{rL}(T_r)$ είναι συναρτήσεις της πίεσης ως προς την ανηγμένη θερμοκρασία η αναλυτική μορφή των οποίων παρουσιάζεται στον Πετρόπουλο (2003). Ειδικά για την $P_{rs}(T_r)$ σημειώνεται ότι πρόκειται για την σχέση (3-3) διατυπωμένη ως εξής: $P_{rs}(T_r) = Ps(T_r)/P^{*}$.

<u>για το βαρύ ύδωρ</u>

Σε αυτό εδώ το σημείο παρουσιάζονται μόνο οι τροποποιήσεις πού χρειάζεται να γίνουν στην διατύπωση IFC-1967, ώστε από αυτήν να υπολογίζονται προσεγγίσεις

ικανοποιητικής ακρίβειας για την πυκνότητα τον βαρέος ύδατος. Συγκεκριμένα οι τροποποιήσεις έχουν ω ς εξής:

- Η συνολική περιοχή ισχύος της διατύπωσης IFC-1967 καλύπτει την περιοχή πιέσεων $0 \leq P \leq 1000$bar και την περιοχή θερμοκρασιών $0 \leq T \leq 1073.15$K. Με σκοπό την ταύτιση της περιοχής ισχύος της τροποποιημένης διατύπωσης IFC-1967 με την κοινή περιοχή ισχύος των εξισώσεων για τις θερμοφυσικές ιδιότητες του βαρέος ύδατος όπου Tmax = 1073.15 K τίθεται Tmax = 873.15 (η πίεση Pmax παραμένει ως έχει).

- Η συνολική περιοχή ισχύος της διατύπωσης IFC-1967 διαχωρίζεται σε έξι υποπεριοχές, οι οποίες καθορίζονται στον Πίνακα 3.1 και απεικονίζονται στα Διαγράμματα 3.1 και 3.2. Γίνεται δεκτό ότι οι θερμοδυναμικές καταστάσεις του βαρέος ύδατος μπορούν να διαχωρισθούν σε ανάλογες υποπεριοχές παρόμοιες με αυτές του ελαφρού ύδατος. Ένας τέτοιος διαχωρισμός είναι βέβαια αυθαίρετος πλην όμως θεμιτός δεδομένης της ομοιότητας στη θερμοδυναμική συμπεριφορά των δύο αυτών ουσιών.

- Αντικαθίσταται ο κρίσιμος ειδικός όγκος τον ελαφρού ύδατος με εκείνον του βαρέος ύδατος. Δηλαδή όπου $\nu^*_{LW.} = 0.00317\ m^3kg^{-1}$ τίθεται $\nu^*_{H\,W} = 0.00279\ m^3kg^{-1}$

- Αντικαθίστανται τα κρίσιμα μεγέθη πιέσεως και θερμοκρασίας τον ελαφρού ύδατος με τα αντίστοιχα μεγέθη του βαρέος ύδατος. Δηλαδή όπου $P^*_{LW} = 221.2$ bar τίθεται $P^*_{HW} = 216.6$ bar και όπου $T^*_{LW} = 647.3$ K τίθεται $T^*_{HW} = 643.89$ K

- Η τιμή της σταθερής $P_{r2} = (Pmax/P^*)_{LW}$ τροποποιείται σε $P_{r2} = (Pmax/P^*)_{HW}$ Δηλαδή $P_{r2} = 4.616805171$

- Η τιμή της σταθερής $T_{r3} = (Tmax/T^*)_{LW}$ τροποποιείται σε $T_{r3} = (Tmax/T^*)_{HW}$ Δηλαδή $T_{r3} = 1.356054606$

- Αντικαθίστανται οι τιμές της πίεσης και της θερμοκρασίας του ελαφρού ύδατος στο τριπλό σημείο με εκείνες του βαρέος ύδατος. Δηλαδή όπου $Pt_{LW.} = 0.006112$ bar τίθεται $Pt_{HW} = 0.06601$ bar, όπου $Tt_{LW} = 273.16$ K τίθεται $Tt_{H.W.} = 276.95$ K. [Αυτό ισοδύναμα σημαίνει ότι $Trt = (Tt/T^*)_{HW} = 4.30120051 \times 10^{-1}$ -βλέπε στον Πίνακα (3.1)]

- Αντικαθίσταται η εξίσωση $P_{rs}=P_{rs}(T_r)$ της διατύπωσης IFC-1967 με την εξίσωση κορεσμού του βαρέος ύδατος ως εξής: $P_{rs}(T_r) = Ps(T)/P^*_{HW}$ όπου το μέγεθος Ps(T) υπολογίζεται από τη σχέση (3-4). Οι τιμές των σταθερών T_{r1}, και T_{r2} παραμένουν ως είχαν:
  $T_{r1} = 9.626911787\times10^{-1}, T_{r2} = 1.333462073$

3.6 Υπολογισμός της θερμοκρασίας κορεσμού

Στην περίπτωση που ζητείται να υπολογισθούν οι θερμοφυσικές ιδιότητες του ελαφρού ή του βαρέος ύδατος ύδατος στην κατάσταση κορεσμού και το μόνο σχετικό δεδομένο είναι η πίεση κορεσμού Ps, τότε πρέπει να επιλυθεί η εξίσωση κορεσμού (3-3) ή (3-4) αντίστοιχα, ως προς τη θερμοκρασία κορεσμού Ts, προκειμένου να υπολογιστεί ακριβώς το ζεύγος πίεσης και θερμοκρασίας για το οποίο πρέπει να γίνουν οι υπολογισμοί των θερμοφυσικών ιδιοτήτων του ελαφρού ή του βαρέος ύδατος.

Ο Πετρόπουλος (2003) για την λύση της εξίσωσης αυτής ως προς τη θερμοκρασία χρησιμοποιεί και πάλι την αριθμητική μέθοδο Newton - Raphson. Η απαραίτητη για τη μέθοδο αρχική τιμή της θερμοκρασίας κορεσμού που χρειάζεται η Newton - Raphson όταν εκκινεί, μπορεί να δίνεται ως συνάρτηση της πίεσης κορεσμού από κατάλληλες συσχετίσεις οι οποίες δίνονται από τον Πετρόπουλο (2003) και έχουν ως εξής:

<u>για το ελαφρύ ύδωρ</u>

$$T_s = a + bP_s^c \qquad (3\text{-}5)$$

με a= 219.88851, b = 146.70687 και c = 0.19753

Η συσχέτιση της μορφής Ts = Ts(Ps) είναι κατάλληλη ώστε να τροφοδοτήσει την χρησιμοποιούμενη αριθμητική μέθοδο Newton - Raphson με την απαιτούμενη αρχική τιμή της θερμοκρασίας κορεσμού του ελαφρού ύδατος με σκοπό να επιτευχθεί ο εύκολος και γρήγορος υπολογισμός-της από την εξίσωση κορεσμού (3-3). Επισημαίνεται ότι η συσχέτιση (3-5) έχει τη μορφή αντίστοιχων συσχετίσεων, οι οποίες παρουσιάσθηκαν από τον Garland (1988, 1989 και 1992).

<u>για το βαρύ ύδωρ</u>

$$T_s = a + bP_s^c \qquad (3\text{-}6)$$

με a = 226.1602382, b = 146.0981428 και c = 0.1940222581

Η συσχέτιση της μορφής Ts = Ts(Ps) είναι κατάλληλη ώστε να τροφοδοτήσει την χρησιμοποιούμενη αριθμητική μέθοδο Newton - Raphson με την απαιτούμενη αρχική τιμή της θερμοκρασίας κορεσμού του βαρέος ύδατος με σκοπό να επιτευχθεί ο εύκολος και γρήγορος υπολογισμός-της από την εξίσωση κορεσμού (3-4).

3.7 Χρήση των εξισώσεων σε κώδικα υπολογισμών

Οι πιοπάνω εξισώσεις, όπως παρουσιάσθηκαν, για τις περιοχές πιέσεων και θερμοκρασιών για τις οποίες ισχύουν, μπορούν να χρησιμοποιηθούν σε κώδικες για τον υπολογισμό των θερμοδυναμικών ιδιοτήτων και των ιδιοτήτων μεταφοράς του ελαφρού και του βαρέος ύδατος και των αντίστοιχων ατμών. Παρόμοιοι κώδικες υπάρχουν αρκετοί, ένας τέτοιος είναι για παράδειγμα ο WASP (WAter and Steam Properties) που παρουσίασε ο Hendricks (1973) ή αυτός του Haar (1984) για το ελαφρύ νερό. Ο Πετρόπουλος (2003) χρησιμοποιεί τις εξισώσεις που παρουσιάσθηκαν για την ανάπτυξη δύο κωδίκων υπολογισμών σε γλώσσα FORTRAN

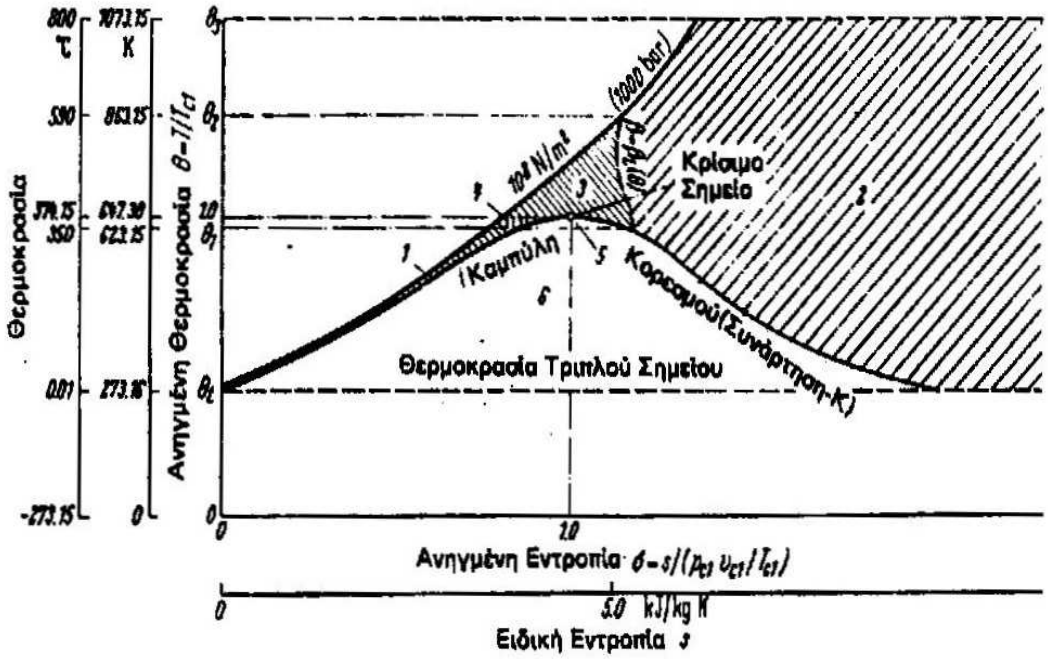
(α) του κώδικα LIGHT_WASP (LIGHT WAter and Steam Properties)

και

(β) του κώδικα HEAVY_WASP (HEAVY WAter and Steam Properties)

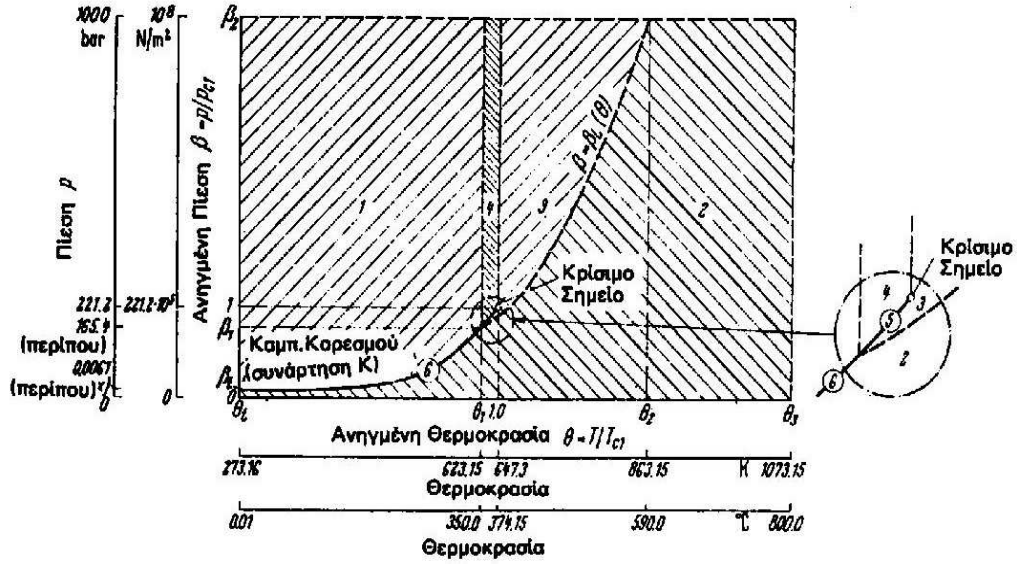
Οι κώδικες αυτοί δέχονται ως μεταβλητές εισόδου την πίεση (σε bar) και τη θερμοκρασία (σε ^{o}C). Οι ιδιότητες που γίνονται διαθέσιμες σε κάθε υπολογισμό ως αποτελέσματα (έξοδοι), είναι: πίεση, θερμοκρασία, πυκνότητα, εντροπία, ενθαλπία, ειδικές θερμοχωρητικότητες, ταχύτητα του ήχου, συνεκτικότητα, θερμική αγωγιμότητα, επιφανειακή τάση και σταθερά Laplace. Οι κώδικες κατασκευάσθηκαν ώστε να αποτελούνται από επιμέρους τμήματα (functions) που επιλέγει να τα καλέσει ο χρήστης ανάλογα με το αν είναι απαραίτητα για τους υπολογισμούς του. Οι κώδικες είναι σχεδιασμένοι να λειτουργούν σαν συνάρτηση εντός του κύριου προγράμματος του χρήστη. Επιπλέον οι κώδικες έχουν την δυνατότητα επίτευξης υπολογισμών μετασταθών καταστάσεων. Στο επόμενο 4^{o} Κεφάλαιο δίνονται λεπτομέρειες σχετικά με τις οδηγίες χρήσης των κωδίκων αυτών. Ένας χρήστης με περιορισμένη εμπειρία δεν έχει καμία δυσκολία να ακολουθήσει τις οδηγίες χρήσης που δίνονται.

ΔΙΑΓΡΑΜΜΑΤΑ ΤΟΥ 3^{ου} ΚΕΦΑΛΑΙΟΥ



Διάγραμμα 3.1

Υποπεριοχές διαγράμματος T-S ελαφρού ύδατος



Διάγραμμα 3.2

Υποπεριοχές διαγράμματος P-V ελαφρού ύδατος

ΚΕΦΑΛΑΙΟ 4
ΔΟΜΗ ΚΩΔΙΚΩΝ ΓΙΑ ΤΟΝ ΥΠΟΛΟΓΙΣΜΟ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΥΔΑΤΟΣ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ

4.1 Εισαγωγή

Στις παραγράφους που ακολουθούν γίνεται η παρουσίαση των αριθμητικών κωδίκων με τα ονόματα light_wasp (light water and steam properties) και heavy_wasp (heavy water and steam properties), σε επίπεδο δομής και λογικής λειτουργίας. Οι κώδικες αυτοί δημιουργήθηκαν σε γλώσσα προγραμματισμού FORTRAN, στα πλαίσια της Διδακτορικής Διατριβής του Πετρόπουλου (2003), με σκοπό να εξυπηρετήσουν υπολογισμούς για τις θερμοδυναμικές ιδιότητες και τις ιδιότητες μεταφοράς του ελαφρού ύδατος και του βαρέος ύδατος αντίστοιχα. Το επίπεδο ακρίβειας των υπολογισμών είναι κατάλληλο για βιομηχανική χρήση, προκειμένου να χρησιμοποιηθεί για υπολογισμούς σε προβλήματα θερμοϋδραυλικής ανάλυσης Πυρηνικών Αντιδραστήρων Ισχύος. Το σύνολο των αναγκαίων συσχετίσεων και των άλλων εξισώσεων που υλοποιούνται προγραμματιστικά παρουσιάσθηκαν ήδη με λεπτομέρεια στο 3° Κεφάλαιο σε συνδυασμό με όσα έχουν παρουσιασθεί και στον Πετρόπουλο (1991, 2003).

Υπενθυμίζεται ότι οι κώδικες δέχονται ως είσοδο την πίεση ή/και την θερμοκρασία. Η έξοδος παρέχει τις τιμές του ειδικού όγκου, της ενθαλπίας, της εντροπίας, των ειδικών θερμοχωρητικοτήτων υπό σταθερή πίεση και υπό σταθερό όγκο, της ταχύτητας του ήχου, της ισοθερμοκρασιακής συμπιεστότητας, του συντελεστή Joule - Thomson, της συνεκτικότητας, της θερμικής αγωγιμότητας και της επιφανειακής τάσης. Αν ο χρήστης των κωδίκων επιθυμεί τις ιδιότητες τον ελαφρού ή του βαρέος ύδατος σε κατάσταση κορεσμού για μια πίεση P, πρέπει να θέσει στην είσοδο όπου θερμοκρασία T την τιμή 0 (μηδέν). Αντίστροφα αν ο χρήστης επιθυμεί το ίδιο για μια θερμοκρασία T. πρέπει να θέσει στην είσοδο όπου πίεση P την τιμή 0 (μηδέν).

Ακολουθεί μια διεξοδική παρουσίαση των γενικών χαρακτηριστικών των κωδίκων και των αναγκαίων οδηγιών χρήσεως. Οι οδηγίες χρήσεως συμπληρώνονται με ορισμένες

υποδείξεις σχετικά με τους τρόπους με τους οποίους ο χρήστης μπορεί να αντιμετωπίσει τα συνηθέστερα λάθη των κωδίκων. Όλα τα υποπρογράμματα εξετάζονται στη συνέχεια κατά ομάδες ανάλογα και με τις λειτουργίες που αυτά διεκπεραιώνουν και παρουσιάζεται ο τρόπος λειτουργίας των κωδίκων, στην ουσία δηλαδή η ροή της εκτέλεσης ανάλογα και με τα ζητούμενα που καθορίζει ο χρήστης. Η συνολική παρουσίαση συνοδεύεται από συγκεντρωτικούς πίνακες πού χρησιμεύουν ως περιληπτική αναφορά στα κυριότερα σημεία των κωδίκων. Τέτοιοι πίνακες συμβάλλουν όχι μόνο στην καλύτερη κατανόησή τους, αλλά και στην γρήγορη αναδρομή όταν ζητείται κάποια λεπτομέρεια που ενδιαφέρει.

Σημειώνεται ότι τόσο στο περιβάλλον προγραμματισμού MATLAB, όσο και σε αυτό του SCILAB διατίθενται μία και μόνο μορφή υποπρογράμματος, αυτό της συνάρτησης (function), σε αντίθεση με τη FORTRAN ή άλλες γλώσσες που διαθέτουν περισσότερες από μία μορφές. Στην περίπτωση που χρειάζεται να υπάρχει ένα κείμενο εντολών (script), οι οποίες πρέπει να επαναλαμβάνονται από καιρό σε καιρό σε διάφορα σημεία ενός κώδικα MATLAB ή SCILAB, αυτό μπορεί να τοποθετείται σε χωριστό αρχείο του κώδικα και να καλείται όταν χρειάζεται. Επίσης, σημειώνεται ότι στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB είναι αποδεκτοί ως χαρακτήρες για τις παντός τύπου μεταβλητές και τα "κεφαλαία" και τα "πεζά". Μία μεταβλητή με όνομα σε "κεφαλαία" είναι διαφορετική από άλλη μεταβλητή με το ίδιο όνομα σε "πεζά". Αυτό γενικά δεν ισχύει στην FORTRAN, όπου δύο τέτοιες μεταβλητές ταυτίζονται. Παρόλαυτά, όλες οι εντολές MATLAB ή SCILAB συντάσσονται με "πεζούς" χαρακτήρες. Επιπλέον όλα τα ονόματα των υποπρογραμμάτων πρέπει επίσης να είναι σε "πεζά".

4.2 Γενικά χαρακτηριστικά κωδίκων

Οι κώδικες light_wasp και heavy_wasp για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού ύδατος και του βαρέος ύδατος είναι στην ουσία μια ομάδα υποπρογραμμάτων σχεδιασμένη για να χρησιμοποιείται ως υποπρόγραμμα στο πρόγραμμα του χρήστη. Τα υποπρογράμματα πού συγκροτούν την ομάδα αυτή διακρίνονται σε:

α) Υποπρογράμματα εισόδου - εξόδου·

β) Υποπρογράμματα υπολογισμού της θερμοκρασίας κορεσμού.

γ) Υποπρογράμματα υπολογισμού του ειδικού όγκου

δ) Υποπρογράμματα υπολογισμού θερμοδυναμικών ιδιοτήτων.

ε) Υποπρογράμματα υπολογισμού ιδιοτήτων μεταφοράς.

στ) Υποπρογράμματα γενικών μαθηματικών υπολογισμών.

ζ) Λοιπά υποπρογράμματα επικουρικής φύσεως.

Τα υποπρογράμματα αυτά μεταφέρθηκαν από την πρωτότυπη μορφή τους σε γλώσσα προγραμματισμού FORTRAN σε περιβάλλοντα προγραμματισμού MATLAB και SCILAB και υλοποιήθηκαν προγραμματιστικά με τη μορφή **συναρτήσεων** (MATLAB / SCILAB functions) και με τη μορφή **κειμένων εντολών** (MATLAB / SCILAB scripts). Σημειώνεται ότι όλα τα υποπρογράμματα που συγκροτούν τους κώδικες βρίσκονται συγκεντρωμένα στο Παράρτημα της Διπλωματικής Εργασίας.

Συνοπτικά η λειτουργία των κωδίκων μπορεί να περιγραφεί με τα ακόλουθα κύρια βήματα:

- Διαβάζεται η πίεση και η θερμοκρασία εισόδου σε bar και °C αντίστοιχα, καθώς και ένας ειδικός κλειδάριθμος (μία ακέραια μεταβλητή), η τιμή του οποίου καθορίζει για ποιες από τις διαθέσιμες θερμοφυσικές ιδιότητες θα γίνουν υπολογισμοί.
- Γίνεται έλεγχος αν η πίεση και η θερμοκρασία εισόδου είναι συμβατές με την περιοχή ισχύος των υλοποιούμενων εξισώσεων για τις ιδιότητες.
- Υπολογίζεται η θερμοκρασία κορεσμού για την πίεση εισόδου, εκτός, φυσικά αν η τελευταία είναι μεγαλύτερη από την κρίσιμη πίεση. προκειμένου να διαπιστωθεί αν οι συνθήκες εισόδου είναι κοντά στην κατάσταση κορεσμού.
- Υπολογίζεται ο ειδικός όγκος για την πίεση και τη θερμοκρασία εισόδου.
- Υπολογίζονται οι βασικές παράμετροι της χρησιμοποιούμενης καταστατικής εξίσωσης, δεδομένων της θερμοκρασίας εισόδου και του υπολογισθέντος ειδικού όγκου.
- Υπολογίζονται οι ζητούμενες με τον ειδικό κλειδάριθμο θερμοφυσικές ιδιότητες.

- Παρέχονται στον χρήστη τα αποτελέσματα.

Ο Πίνακας 4.1 παρουσιάζει τα σύμβολα, την ονοματολογία και τις μονάδες των βασικών μεταβλητών των κωδίκων light_wasp και heavy_wasp σε αλφαβητική σειρά. Ως βασικές μεταβλητές εννοούνται εδώ κυρίως τα καταστατικά μεγέθη και οι θερμοφυσικές ιδιότητες του ελαφρού / βαρέος ύδατος καθώς και ορισμένες άλλες παράμετροι με τις οποίες ο χρήστης έρχεται σε άμεση επαφή. Τα σύμβολα των μεταβλητών συνοδεύονται από τις διαστάσεις τους σε μονάδες στις οποίες γίνεται η είσοδος ή η έξοδος. Σημειώνεται πάντως ότι οι κώδικες χρησιμοποιούν εσωτερικά σε αρκετά σημεία τους και διαφορετικές μονάδες για το ίδιο μέγεθος ή ιδιότητα με σκοπό να εξυπηρετήσουν ανάγκες τοπικών, εσωτερικών υπολογισμών.

4.3 Οδηγίες χρήσεως

Στην παράγραφο αυτή παρουσιάζονται οι απαραίτητες οδηγίες για τη σωστή κλήση και εκμετάλλευση των κωδίκων light_wasp και heavy_wasp από προγράμματα εφαρμογής του χρήστη. Η τυπική επικοινωνία του προγράμματος του χρήστη με τους κώδικες αποκαθίσταται με την ακόλουθη ομάδα εντολών, οι οποίες περιέχουν τις μεταβλητές που αντιπροσωπεύουν τις παραμέτρους εισόδου / εξόδου. Δεδομένων των κοινών χαρακτηριστικών των περιβαλλόντων προγραμματισμού MATLAB και SCILAB η ομάδα εντολών που χρησιμοποιείται και στις δύο περιπτώσεις έχει σχεδόν ίδια μορφή. Εδώ παρουσιάζεται η ομάδα εντολών του περιβάλλοντος MATLAB.

για το ελαφρύ νερό

```
constants;
transient;
[P,T,TS,SVF,SVG,JR,IS]=light_wasp(JS,JP,P,T);
if JR==0
      return
end
```

και για το βαρύ νερό

```
constants;
```

```
transient;
[P,T,TS,SVF,SVG,JR,IS]=heavy_wasp(JS,JP,P,T);
if JR==0
      return
end
```

Είναι:

JS, JP διακόπτες, ο ρόλος των οποίων περιγράφεται στη συνέχεια (μεταβλητές εισόδου)

P πίεση σε bar (μεταβλητή εισόδου)

T θερμοκρασία σε K (μεταβλητή εισόδου)

TS θερμοκρασία κορεσμού στην πίεση P (σε K -μεταβλητή εξόδού)

SVF ειδικός όγκος υγρής ή υπερκρίσιμης φάσης (σε m^3/kg -μεταβλητή εξόδου)

SVG ειδικός όγκος ατμώδους ή υπερκρίσιμης φάσης (σε m^3/kg -μεταβλητή εξόδου)

JR, IS δείκτες (μεταβλητές εξόδου -παίρνουν τιμές με τρόπο που περιγράφεται στη συνέχεια)

Το script transient – το οποίο καλείται με την εντολή transient; - περιέχει τις υπόλοιπες μεταβλητές εξόδου. Οι μεταβλητές αυτές παρουσιάζονται συγκεντρωτικά στον Πίνακα 4.1.

Το script constants – το οποίο καλείται με την εντολή constants; - περιέχει μεταβλητές των κωδίκων που έχουν σταθερές τιμές.

Για την επιτυχή κλήση των κωδίκων πρέπει να τηρούνται δύο προϋποθέσεις:

1. Η κλήση του script transient πρέπει να γίνεται στο κυρίως πρόγραμμα του χρήστη ή στην υπορουτίνα του, η οποία καλεί τον κώδικα. Όλες οι μεταβλητές που περιέχονται στις εντολές του script εννοούνται διπλής ακρίβειας (DOUBLE PRECISION), κατά τη σύμβαση των προγραμματιστικών περιβαλλόντων MATLAB και SCILAB. Οπωσδήποτε επιτρέπεται στο χρήστη να αλλάξει τα ονόματα των μεταβλητών που εμφανίζονται στην εντολή [P,T,TS,SVF,SVG,JR,IS]=light_wasp(JS,JP,P,T) ή στην

εντολή [P,T,TS,SVF,SVG,JR,IS]=heavy_wasp(JS,JP,P,T) κατά τον τρόπο που αυτός νομίζει ότι τον εξυπηρετεί. Προσοχή χρειάζεται ώστε να αποφευχθεί η χρησιμοποίηση των ονομάτων των μεταβλητών που εμφανίζονται στις εντολές των script ως δείκτες για στοιχεία πινάκων.

2. Οι διακόπτες JS και JP, τις τιμές των οποίων παρέχει ο χρήστης, πληροφορούν τους κώδικες σχετικά με το ποιες μεταβλητές θα χρησιμεύσουν ως είσοδος και με το ποιες ιδιότητες ζητείται να αποτελέσουν την έξοδο. Πρέπει να πάρουν τις κατάλληλες αρχικές τιμές. Ο διακόπτης JS καθορίζει ποιες από τις μεταβλητές θερμοκρασία, πίεση, ειδικός όγκος, ενθαλπία ή εντροπία, ποιόν συνδυασμό τους ανά δύο θέλει ο χρήστης σαν είσοδο. Στην παρουσιαζόμενη έκδοση τον κώδικα, η μόνη τιμή που επιτρέπεται να πάρει ο διακόπτης JS είναι η μονάδα ("1"), πού αντιστοιχεί στα ζεύγη εισόδου (P,T), (P,0) και (0,T). Ο διακόπτης JP καθορίζει ποιες ιδιότητες ζητάει ο χρήστης να υπολογισθούν από τον κώδικα. Ο διακόπτης αυτός μπορεί να πάρει τις ακέραιες τιμές από 0 έως και 63. Οι τιμές αυτές αντιπροσωπεύουν κάποιο άθροισμα των αριθμών 0, 2, 4, 8, 16 και 32 –δηλαδή ισοδύναμα των αριθμών 0, 2^0, 2^1, 2^2, 2^3, 2^4 και 2^5. Κάθε τέτοιο δυαδικό άθροισμα εξυπηρετεί με μοναδικό τρόπο τον προσδιορισμό του συνδυασμού των ιδιοτήτων που ζητούνται. Ο Πίνακας 4.2 συνοψίζει ποιες ιδιότητες υπολογίζονται αν ο διακόπτης JP πάρει τις τιμές 0, 2^0, 2^1, 2^2, 2^3, 2^4 και 2^5.

   Για να γίνει περισσότερο κατανοητή η λειτουργία του διακόπτη JP δίνονται και τα ακόλουθα **παραδείγματα:**

   Έστω JP = 63 (ισοδύναμα JP = 0 + 1 + 2 + 4 + 8 + 16 + 32 = 63) τότε φανερό είναι ότι ο χρήστης ζητά τον υπολογισμό όλων των ιδιοτήτων που μπορεί να τον δώσει ο κώδικας.

   Έστω JP = 18 (ισοδύναμα JP = 0 + 2 + 16 = 18) τότε ο χρήστης ζητά μόνο τον υπολογισμό, της ενθαλπίας, της εντροπίας και της επιφανειακής τάσης. Είναι φανερό ότι τα αθροίσματα-παραδείγματα 63 και 18 μπορούν να προκύψουν μόνο με μοναδικούς συνδυασμούς των διαθέσιμων αριθμών (0, 1, 2, 4, 8, 16, 32), πράγμα που εξασφαλίζει την σαφήνεια της επιλογής του χρήστη, όσο αφορά στις ιδιότητες του ελαφρού ύδατος που επιθυμεί να υπολογίσει.

Ο διακόπτης JP βοηθάει στην αύξηση της ταχύτητας με την οποία οι κώδικες light_wasp και heavy_wasp υπολογίζουν τις ζητούμενες θερμοφυσικές ιδιότητες του ελαφρού ύδατος και του βαρέος ύδατος αφού από την αρχή της εκτελέσεως πληροφορεί ποια από τα τμήματα των κωδίκων θα παραμείνουν ανενεργά. Για μικρές τιμές τον JP αυτό σημαίνει την αδρανοποίηση μέχρι περίπου και του ενός τρίτου των εντολών τους.

Εκτός από τους διακόπτες JS και JP και τις άλλες μεταβλητές εισόδου οι κώδικες περιέχουν στη λίστα των παραμέτρων τους και τους δείκτες JR και IS, οι οποίοι παίρνουν τιμές μόνο μετά από μια επιτυχημένη κλήση και εκτέλεση του κώδικα.

Ο δείκτης JR αποδίδει την περιοχή του διαγράμματος PT στην οποία βρίσκεται η πίεση και η θερμοκρασία εισόδου σύμφωνα και με το Διάγραμμα 3.2. Οι επιτρεπόμενες τιμές για το δείκτη JR είναι 1, 2, 3, 4, 5 και 6. Στην απίθανη περίπτωση που οι κώδικες επιστρέψουν για τον δείκτη JR την τιμή 0 (μηδέν), σημαίνει ότι έχει αποτύχει για κάποιο λόγο να καθορισθεί η περιοχή του διαγράμματος PT στην οποία βρίσκεται η πίεση και η θερμοκρασία εισόδου που καθόρισε ο χρήστης. Τυπώνεται τότε σχετικό μήνυμα και η εκτέλεση σταματάει.

Ο δείκτης IS πληροφορεί το χρήστη σε τι σχέση βρίσκονται η πίεση και η θερμοκρασία εισόδου που καθόρισε, ως προς την κατάσταση κορεσμού για την πίεση εισόδου. Παίρνει τις τιμές 0,1 και 2. Ο Πίνακας 4.3 εξηγεί αναλυτικά τι σημαίνει κάθε τιμή του δείκτη IS.

4.4 Αντιμετώπιση λαθών (troubleshooting)

Ο τρόπος με τον οποίο είναι κατασκευασμένοι οι κώδικες light_wasp και heavy_wasp αντιμετωπίζουν τα συνήθη λάθη που τυχόν συμβαίνουν κατά την κλήση τους. Αυτά τα λάθη ανιχνεύονται και μπορούν να διορθωθούν σχετικά εύκολα. Αναλυτικότερα:

1. Αν δεν καθορισθεί σωστά ο διακόπτης JS (πρέπει JS=1) τυπώνεται σχετικό μήνυμα στην εκ ταυτότητας μονάδα εξόδου του χρήστη, στον υπολογιστή που αυτός εργάζεται και η εκτέλεση σταματάει.
2. Αν δεν καθορισθεί σωστά ο διακόπτης JP (πρέπει ο JP να είναι μεταξύ του 0 και τον 63) τυπώνεται σχετικό μήνυμα στην εκ ταυτότητας μονάδα εξόδου του χρήστη, όπως προηγούμενα και η εκτέλεση σταματάει.

3. Αν οι τιμές των P,T βρίσκονται έξω από την περιοχή ισχύος τυπώνεται ομοίως μήνυμα και η εκτέλεση σταματάει.
4. Αν ζητείται μια κατάσταση κορεσμού αλλά από λάθος η πίεση ή η θερμοκρασία εισόδου είναι υπερκρίσιμες τυπώνεται ομοίως μήνυμα και η εκτέλεση σταματάει.
5. Αν η σύνταξη της απαραίτητης για την κλήση των κωδίκων ομάδας εντολών, είναι εσφαλμένη τότε καταλαβαίνει κανείς ότι υπάρχει μια ποικιλία λαθών που μπορεί να συμβεί, τα περισσότερα των οποίων είναι βέβαια μοιραία (fatal) και η εκτέλεση σταματάει. Συνήθως τότε τυπώνεται και κατάλληλο μήνυμα λάθους που προέρχεται όμως από τον ίδιο τον υπολογιστή και όχι από τον κώδικα.

Λοιπά μηνύματα που μπορεί να εμφανισθούν κατά την εκτέλεση των κωδίκων είναι αυτοεξηγούμενα.

4.5 Υποπρογράμματα

Όπως ήδη αναφέρθηκε τα υποπρογράμματα των κωδίκων light_wasp και heavy_wasp μπορούν να διακριθούν σε υποπρογράμματα:

- εισόδου – εξόδου,
- υπολογισμού της θερμοκρασίας κορεσμού,
- υπολογισμού του ειδικού όγκου δεδομένης της πίεσης και της θερμοκρασίας εισόδου,
- υπολογισμού θερμοδυναμικών ιδιοτήτων,
- υπολογισμού ιδιοτήτων μεταφοράς,
- μαθηματικών υπολογισμών και τέλος
- άλλα επικουρικά υποπρογράμματα.

Στη συνέχεια θα περιγραφούν με συνοπτικό τρόπο όλα τα υποπρογράμματα των κωδίκων κατατασσόμενα κατά την πιοπάνω κατηγοριοποίηση.

4.5.1 Υποπρογράμματα εισόδου - εξόδου

Τα υποπρογράμματα εισόδου - εξόδου είναι ονομαστικά οι functions light_wasp (και heavy_wasp), checkpt και οι συνεργαζόμενες functions fps και prl.

4.5.1.1 Υποπρόγραμμα (light_ / heavy_) wasp

Είναι το υποπρόγραμμα που πρέπει να κληθεί από το πρόγραμμα του χρήστη αν αυτός θέλει να χρησιμοποιήσει έναν κώδικα. Ο τρόπος με τον οποίο γίνεται αυτή η κλήση ήδη παρουσιάσθηκε. Το υποπρόγραμμα αποτελείται από μία ομάδα εντολών που φροντίζει κυρίως για την ομαλή είσοδο στον κώδικα και την ομαλή έξοδο από αυτόν. Εκεί ελέγχεται η ορθότητα των τιμών των διακοπτών JS και JP, δίνεται η εντολή για τον έλεγχο της ορθότητας της πίεσης και της θερμοκρασίας εισόδου, αρχικοποιούνται οι τιμές όλων των ιδιοτήτων στο μηδέν και συντονίζονται οι κυριότερες λειτουργίες του κώδικα. Κατά τη διαδικασία εξόδου και μέσω της λίστας παραμέτρων το υποπρόγραμμα …wasp πληροφορεί το πρόγραμμα του χρήστη για τις τιμές που παίρνει ο ειδικός όγκος του υγρού (SVF σε m^3/kg) ή/και ο ειδικός όγκος του ατμού (SVG σε m^3/kg) του ελαφρού ή του βαρέος ύδατος για το ζεύγος πιέσεως - θερμοκρασίας, στο οποίο ζητήθηκε να γίνουν υπολογισμοί.

4.5.1.2 Υποπρόγραμμα checkpt

Όπως φαίνεται και από το όνομά της πρόκειται για την function, στην οποία η function …wasp αναθέτει τον έλεγχο της πίεσης και της θερμοκρασίας εισόδου. Το υποπρόγραμμα μετατρέπει τις μονάδες της πίεσης και της θερμοκρασίας από bar και °C σε MPa και K αντίστοιχα, προκειμένου να διευκολύνονται οι υπολογισμοί. Υπολογίζεται η πίεση κορεσμού αν το μόνο δεδομένο είναι η θερμοκρασία κορεσμού TS με χρήση της συνάρτησης fps. Υπολογίζεται αντίστροφα η θερμοκρασία κορεσμού TS αν το μόνο δεδομένο είναι η πίεση κορεσμού, με χρήση των συναρτήσεων fps, dfps, fts και της μαθηματικής συνάρτησης newton. Αποφασίζεται επίσης εκεί η τιμή της μεταβλητής εξόδου JR σύμφωνα με τα όσα αναφέρονται στον Πίνακα 3.1 και με τη βοήθεια της συνάρτησης prl, καθώς και η τιμή της μεταβλητής εξόδου IS σύμφωνα με όσα αναφέρονται στον Πίνακα 4.3.

4.5.1.3 Υποπρόγραμμα fps

Είναι η υλοποίηση της εξίσωσης κορεσμού του ελαφρού ύδατος ή του βαρέος ύδατος, η οποία χρησιμοποιείται από το υποπρόγραμμα checkpt για τον υπολογισμό της πίεσης κορεσμού (μεταβλητή FPS) σε MPa δεδομένης της θερμοκρασίας κορεσμού TS σε K.

4.5.1.4 Υποπρόγραμμα prl

Είναι η υλοποίηση της εξίσωσης που αναπαριστά το όριο μεταξύ των υποπεριοχών όπου JR = 2 και JR = 3 για καταστάσεις της υπερκρίσιμης φάσης. Υπολογίζεται η μεταβλητή PRL που είναι η ανηγμένη πίεση. Χρησιμοποιείται από το υποπρόγραμμα checkpt, προκειμένου να αποφασισθεί η τιμή του δείκτη JR.

4.5.2 Υποπρογράμματα υπολογισμού θερμοκρασίας κορεσμού

Η θερμοκρασία κορεσμού TS υπολογίζεται, αν το μόνο δεδομένο είναι η πίεση κορεσμού, με χρήση των συναρτήσεων fps, dfps, fts και της μαθηματικής συνάρτησης newton.

4.5.2.1 Υποπρόγραμμα fts

Είναι η υλοποίηση συσχέτισης που έχει σκοπό τον υπολογισμό αρχικών εκτιμήσεων για τη θερμοκρασία κορεσμού σε K ως συνάρτηση της πίεσης κορεσμού σε bar. Οι εκτιμήσεις αυτές είναι απαραίτητες για την επίλυση της εξίσωσης κορεσμού -όπως αυτή υλοποιείται στη συνάρτηση fps, ως προς τη θερμοκρασία με την αριθμητική μέθοδο Newton - Raphson -όπως αυτή υλοποιείται στο μαθηματική συνάρτηση newton. Υπολογίζεται η μεταβλητή FTS που είναι η θερμοκρασία κορεσμού σε K.

4.5.2.2 Υποπρόγραμμα dfps

Είναι η υλοποίηση της παραγώγου της εξίσωσης κορεσμού ως προς τη θερμοκρασία, απαραίτητη για την επίλυση της εξίσωσης κορεσμού ως προς τη θερμοκρασία κορεσμού TS, όταν το μόνο δεδομένο είναι η πίεση κορεσμού δεδομένου ότι χρησιμοποιείται η αριθμητική μέθοδος Newton - Raphson. Η μέθοδος Newton - Raphson υλοποιείται στην μαθηματική συνάρτηση newton.

4.5.3 Υποπρογράμματα υπολογισμού ειδικού όγκου

Τα υποπρογράμματα υπολογισμού του ειδικού όγκου δεδομένης της πίεσης και της θερμοκρασίας εισόδου είναι ονομαστικά οι συναρτήσεις densf, densg, svlwl, svlwv και qmust και οι συνεργαζόμενες συναρτήσεις fpr4, dfpr4, fpr3, dfpr3, fp και dfpd. Τονίζεται ότι αυτή η ομάδα των υποπρογραμμάτων υπολογίζει τον ειδικό όγκο του ελαφρού ύδατος ή του βαρέος ύδατος ακόμα και ακριβώς στο κρίσιμο σημείο και φυσικά σε όλη την περιοχή που γειτνιάζει με αυτό, παρόλο που η καταστατική εξίσωση

που χρησιμοποιείται στους κώδικες δεν θεωρείται ακριβής γύρω από το ζεύγος (P, T). Αυτό έγινε διότι για το σαφή καθορισμό των ορίων της γειτονιάς του κρίσιμου σημείου για την οποία ο κώδικας δεν ισχύει είναι απαραίτητο να είναι γνωστές όχι μόνο οι οριακές πιέσεις και θερμοκρασίες αλλά και οι οριακές πυκνότητες. Η προγραμματιστική προσπάθεια για τον υπολογισμό της πυκνότητας του ελαφρού ύδατος και του βαρέος ύδατος στο κρίσιμο σημείο απαίτησε τη σύνταξη των υποπρογραμμάτων fpr3, dfpr3, fpr4 και dfpr4.

4.5.3.1 Υποπρόγραμμα densf

Μετά την ολοκλήρωση του ελέγχου στο υποπρόγραμμα checkpt και αν η μεταβλητή JR έχει την τιμή 1, 4, 5 ή 6 το υποπρόγραμμα …wasp αναθέτει στο υποπρόγραμμα densf να υπολογίσει την πυκνότητα του ελαφρού ύδατος DF σε g/cm^3 στις αντίστοιχες της JR υποπεριοχές (υγρή φάση, κορεσμένο υγρό, μέρος της υπερκρίσιμης φάσης).

4.5.3.2 Υποπρόγραμμα densg

Μετά την ολοκλήρωση του ελέγχου στο υποπρόγραμμα checkpt και αν η μεταβλητή JR έχει την τιμή 2, 3, 5 ή 6 το υποπρόγραμμα …wasp αναθέτει στο υποπρόγραμμα densg να υπολογίσει την πυκνότητα του ελαφρού ύδατος ή του βαρέος ύδατος DG σε g/cm^3 στις αντίστοιχες της JR υποπεριοχές (ατμώδης φάση, κορεσμένος ατμός, μέρος της υπερκρίσιμης φάσης).

4.5.3.3 Υποπρόγραμμα svlwl

Το υποπρόγραμμα αυτό καλείται από το υποπρόγραμμα densf και χρησιμεύει για τον υπολογισμό της αρχικής εκτίμησης DL της πυκνότητας σε g/cm^3 στις υποπεριοχές 1, 4, 5, ή 6 χρησιμοποιώντας κυρίως τις σχέσεις που δίνονται για αυτές τις υποπεριοχές από τον Schmidt (1987). Στη συνέχεια, η αρχική εκτίμηση DL χρησιμοποιείται από την αριθμητική μέθοδο Newton Raphson, που εφαρμόζεται για την επίλυση της καταστατικής εξίσωσης της μορφής P = P(ρ,T) ως προς την πυκνότητα του ελαφρού ύδατος ή του βαρέος ύδατος. Η μέθοδος Newton – Raphson υλοποιείται στη μαθηματική συνάρτηση newton.

4.5.3.4 Υποπρόγραμμα fpr4

Είναι η υλοποίηση της εξίσωσης που δίνεται από τον Schmidt (1987, σελ. 196, ποσότητα β_4) και είναι απαραίτητη για τον υπολογισμό της αρχικής εκτίμησης της πυκνότητας DL για τις υποπεριοχές 4 και 5 (για την επίλυσή της ως προς τη πυκνότητα

χρησιμοποιείται η αριθμητική μέθοδος Newton - Raphson). Υπολογίζεται η μεταβλητή FPR4 που είναι η ανηγμένη πίεση Pr στην υποπεριοχή 4.

4.5.3.5 Υποπρόγραμμα dfpr4

Είναι η υλοποίηση της παραγώγου της εξίσωσης της προηγούμενης παραγράφου ως προς την πυκνότητα. Η παράγωγος αυτή είναι απαραίτητη για την επίλυση της εξίσωσης αυτής ως προς την πυκνότητα με την αριθμητική μέθοδο Newton - Raphson.

4.5.3.6 Υποπρόγραμμα svlwv

Το υποπρόγραμμα αυτό καλείται από το υποπρόγραμμα densg και χρησιμεύει για τον υπολογισμό της αρχικής εκτίμησης DV της πυκνότητας σε g/cm^3 στις υποπεριοχές 2, 3, 5. ή 6, χρησιμοποιώντας κυρίως τις σχέσεις για την ποσότητα χ_2 (βλ. και Schmidt, 1986, σελ. 194) και για την ποσότητα β_3 (βλ. και Schmidt, 1986, σελ. 195). Στη συνέχεια, η αρχική εκτίμηση DV χρησιμοποιείται από την αριθμητική μέθοδο Newton - Raphson, που εφαρμόζεται για την επίλυση της καταστατικής εξίσωσης της μορφής $P = P(\rho,T)$ ως προς την πυκνότητα τον ελαφρού ή του βαρέος ύδατος. Η μέθοδος Newton – Raphson υλοποιείται στη μαθηματική συνάρτηση newton.

4.5.3.7 Υποπρόγραμμα fpr3

Είναι η υλοποίηση της εξίσωσης για την ποσότητα β_3 (βλ. και Schmidt, 1986, σελ. 195) απαραίτητης για τον υπολογισμό της αρχικής εκτίμησης της πυκνότητας DV για τις υποπεριοχές 3 και 5 (για την επίλυσή της ως προς τη πυκνότητα χρησιμοποιείται η αριθμητική μέθοδος Newton - Raphson). Υπολογίζεται η μεταβλητή FPR3 που είναι η ανηγμένη πίεση Pr στην υποπεριοχή 3.

4.5.3.8 Υποπρόγραμμα dfpr3

Είναι η υλοποίηση της παραγώγου της εξίσωσης της προηγούμενης παραγράφου ως προς την πυκνότητα. Η παράγωγος αυτή είναι απαραίτητη για την επίλυση της εξίσωσης αυτής ως προς την πυκνότητα με την αριθμητική μέθοδο Newton - Raphson.

4.5.3.9 Υποπρόγραμμα fp

Είναι η υλοποίηση της καταστατικής εξίσωσης $P = P(\rho,T)$ όπως αυτή χρησιμοποιείται από το μαθηματικό υποπρόγραμμα newton προκειμένου να επιλυθεί ως προς την

πυκνότητα με χρήση της αριθμητικής μεθόδου Newton - Raphson. Υπολογίζεται η μεταβλητή FP που είναι η πίεση σε MPa. Για τον υπολογισμό της ποσότητας Q και της μερικής παραγώγου της ως προς την πυκνότητα, οι οποίες είναι απαραίτητες για την εξίσωση P = P(ρ,T) χρησιμοποιείται η συνάρτηση qmust.

4.5.3.10 Υποπρόγραμμα dfpd

Είναι η υλοποίηση της μερικής παραγώγου της καταστατικής εξίσωσης P = P(ρ,T) ως προς την πυκνότητα. Η παράγωγος αυτή είναι απαραίτητη για την επίλυση της καταστατικής εξίσωσης ως προς την πυκνότητα με την αριθμητική μέθοδο Newton - Raphson. Η μέθοδος Newton υλοποιείται στην μαθηματική συνάρτηση newton. Για τον υπολογισμό των ποσοτήτων Q, και των μερικών παραγώγων της, οι οποίες είναι απαραίτητες για τον υπολογισμό της ζητούμενης μερικής παραγώγου της καταστατικής εξίσωσης P = P(ρ,T) ως προς την πυκνότητα χρησιμοποιείται το υποπρόγραμμα qmust.

4.5.3.11 Υποπρόγραμμα qmust

Πρόκειται για ένα από τα βασικότερα υποπρογράμματα του κώδικα. Ασχολείται με τον υπολογισμό της ποσότητας Q, και των μερικών πρώτων και δεύτερων παραγώγων της ως προς τη θερμοκρασία και την πυκνότητα, οι οποίες αποδίδονται από τις μεταβλητές QTD, QDT, Q2DT, Q2D2T και Q2T2D. Επίσης προσδιορίζει την ειδική ελεύθερη εσωτερική ενέργεια αναφοράς ψ_o (μεταβλητή PSI0), την πρώτη και τη δεύτερη παράγωγό της ως προς τη θερμοκρασία (μεταβλητές PSI0T, PSI02T2), και άλλες ποσότητες.

4.5.4 Υποπρογράμματα υπολογισμού θερμοδυναμικών ιδιοτήτων

Τα υποπρογράμματα υπολογισμού των θερμοδυναμικών ιδιοτήτων του ελαφρού ή του βαρέος ύδατος είναι ονομαστικά οι συναρτήσεις total, energy, enthalpy, entropy, shp, shv, shr_sove και ibm_jtc.

4.5.4.1 Υποπρόγραμμα total

Όταν υπολογισθεί η πυκνότητα του ελαφρού ή του βαρέος ύδατος, το υποπρόγραμμα …wasp καλεί το υποπρόγραμμα total, το οποίο με τη σειρά του αναθέτει σε διάφορα μικρότερα υποπρογράμματα -η σύντομη περιγραφή των οποίων ακολουθεί- τον υπολογισμό των ζητούμενων θερμοδυναμικών ιδιοτήτων και ιδιοτήτων μεταφοράς του

ελαφρού ή του βαρέος ύδατος ανάλογα με την τιμή του διακόπτη JP. Πολλές από τις θερμοδυναμικές ιδιότητες του ελαφρού ή του βαρέος ύδατος απαιτούν κοινούς υπολογισμούς διαφόρων ποσοτήτων. Οι υπολογισμοί αυτοί γίνονται από το υποπρόγραμμα qmust και κοινοποιούνται στα υποπρογράμματα που τους χρειάζονται μέσω κατάλληλου script που ονομάζεται transient.

4.5.4.2 Υποπρόγραμμα energy

Υπολογίζει την ειδική εσωτερική ενέργεια U και την ειδική ελεύθερη εσωτερική ενέργεια PSI του ελαφρού ή του βαρέος ύδατος σε kJ/kg.

4.5.4.3 Υποπρόγραμμα enthalpy

Υπολογίζει την ειδική ενθαλπία H του ελαφρού ή του βαρέος ύδατος σε kJ/kg.

4.5.4.4 Υποπρόγραμμα entropy

Υπολογίζει την ειδική εντροπία S του ελαφρού ή του βαρέος ύδατος σε kJ/(kgK).

4.5.4.5 Υποπρόγραμμα shp

Υπολογίζει την ειδική θερμοχωρητικότητα υπό σταθερή πίεση (Specific Heat at Constant Pressure) CP σε kJ/(kgK).

4.5.4.6 Υποπρόγραμμα shv

Υπολογίζει την ειδική θερμοχωρητικότητα υπό σταθερό όγκο (Specific Heat at Constant Volume) CV σε kJ/(kgK).

4.5.4.7 Υποπρόγραμμα shr_sove

Υπολογίζει το λόγο των ειδικών θερμοχωρητικοτήτων (specific heat ratio) GAMMA και την ταχύτητα του ήχου (sonic velocity) A σε m/sec.

4.5.4.8 Υποπρόγραμμα ibm_jtc

Υπολογίζει την ισοθερμοκρασιακή συμπιεστότητα (Isothermal Bulk Modulus) IBM σε MPa^{-1} και τον συντελεστή Joule - Thomson (Joule - Thomson Coefficient) JTC σε K/MPa.

4.5.5 Υποπρογράμματα υπολογισμού ιδιοτήτων μεταφοράς

Τα υποπρογράμματα υπολογισμού των ιδιοτήτων μεταφοράς του ελαφρού ή του βαρέος ύδατος είναι ονομαστικά οι συναρτήσεις viscosity, conductive και tension.

4.5.5.1 Υποπρόγραμμα viscosity

Υπολογίζει τη δυναμική συνεκτικότητα MU του ελαφρού ή του βαρέος ύδατος σε kg/(ms).

4.5.5.2 Υποπρόγραμμα conductive

Υπολογίζει τη θερμική αγωγιμότητα K του ελαφρού ή του βαρέος ύδατος σε W/(mK).

4.5.5.3 Υποπρόγραμμα tension

Υπολογίζει την επιφανειακή τάση SIGMA μεταξύ υγρού και ατμού σε N/m^2.

4.5.6 Υποπρογράμματα μαθηματικών υπολογισμών

Το μοναδικό υποπρόγραμμα μαθηματικών υπολογισμών είναι το υποπρόγραμμα newton. Είναι η υλοποίηση της αριθμητικής μεθόδου Newton - Raphson. Υπολογίζει την πραγματική ρίζα X (μεταβλητή εξόδου) μιας πραγματικής συνάρτησης F (function εισόδου) δεδομένης μιας καλής αρχικής εκτίμησης X (μεταβλητή εισόδου) και της πρώτης μερικής παραγώγου DF (function εισόδου) της F ως προς την X. Μία ρίζα X γίνεται αποδεκτή αν η απόλυτη τιμή της F(X) γίνει μικρότερη ή ίση με EPS (μεταβλητή εισόδου -η EPS βρίσκεται στη γειτονιά τον μηδενός). Εναλλακτικά, μία ρίζα X γίνεται αποδεκτή αν φθάσουν δυο διαδοχικές εκτιμήσεις της να συμφωνούν στα πρώτα NDEC (μεταβλητή εισόδου) δεκαδικά τους ψηφία. ITMAX (μεταβλητή εισόδου) είναι ο μέγιστος επιτρεπόμενος αριθμός επαναλήψεων Newton – Raphson για την εύρεση μιας ρίζας. IER (μεταβλητή εξόδου) είναι μία προειδοποιητική παράμετρος για λάθη. Όταν IER = 1 σημαίνει ότι η ζητούμενη ρίζα δεν μπορεί να βρεθεί επειδή η παράγωγος DF "πλησίασε" πολύ την τιμή μηδέν, τότε στη X δίνεται η τιμή 111111. Όταν IER = 2 σημαίνει ότι η ζητούμενη ρίζα δεν μπορεί να βρεθεί διότι καλύφθηκε ο επιτρεπόμενος αριθμός επαναλήψεων ITMAX, τότε στη X δίνεται η τιμή 222222. Όταν IER = 3 σημαίνει ότι συνέβησαν και τα δύο προηγούμενα λάθη, τότε στη X δίνεται είτε η τιμή 111111 είτε η τιμή 222222.

4.5.7 Άλλα υποπρογράμματα

4.5.7.1 Script initial

Είναι ένα κείμενο εντολών, στο οποίο, ως καλή προγραμματιστική πολιτική, δίνονται αρχικές τιμές 0 σε εκείνες από τις μεταβλητές των κωδίκων που υπολογίζονται ως

μεταβλητές εξόδου. Οι μεταβλητές αυτές είναι global (δηλ. καθολικές) για τους κώδικες.

4.5.7.2 Script constants

Είναι ένα κείμενο εντολών, στο οποίο δίνονται τιμές στις περισσότερες από τις σταθερές που είναι απαραίτητες για τις διάφορες συσχετίσεις που υλοποιούνται στους κώδικες. Οι αντίστοιχες μεταβλητές είναι global (δηλ. καθολικές) για τους κώδικες. Ενδεικτικά αναφέρεται ότι στις σταθερές αυτές συμπεριλαμβάνονται οι μέγιστες και οι ελάχιστες πιέσεις και θερμοκρασίες για τις οποίες λειτουργούν οι κώδικες, τα κρίσιμα καταστατικά μεγέθη και άλλες σταθερές που αφορούν την ουσία "ελαφρύ ύδωρ" ή την ουσία "βαρύ ύδωρ", τα όρια των υποπεριοχών, με βάση τα οποία καθορίζεται η μεταβλητή JR κ.α. Σημειώνεται ότι στο script αυτό δίνονται μόνο σταθερές που κατά κανόνα χρησιμοποιούνται σε περισσότερα από ένα υποπρογράμματα.

4.5.7.3 Script transient

Είναι ένα κείμενο εντολών, το οποίο καλείται κάθε φορά που κάποια από τις μεταβλητές οι οποίες περιέχονται εκεί αλλάζει τιμή. Αυτό γίνεται προκειμένου να ενημερώνονται τα επί μέρους τμήματα του κώδικα για τις αλλαγές αυτές. Οι υπόψιν μεταβλητές είναι global (δηλ. καθολικές) για τους κώδικες.

4.5.7.4 Κυρίως πρόγραμμα example και συγκριτικά αποτελέσματα

Στα Παραρτήματα του 5^{ου} και του 6^{ου} Κεφαλαίου όπου δίνονται τα κείμενα των κωδίκων …wasp για το ελαφρύ και το βαρύ νερό, συμπεριλαμβάνεται, εκτός των υποπρογραμμάτων και ένα κυρίως πρόγραμμα εφαρμογής-παράδειγμα ειδικά για να κατανοήσει ο χρήστης των κωδίκων πως πρέπει να καλεί το υποπρόγραμμα …wasp. Το κυρίως πρόγραμμα example παρέχει και μία στοιχειώδη έξοδο για τις περισσότερες από τις ιδιότητες του ελαφρού ή του βαρέος ύδατος που είναι δυνατόν να υπολογισθούν από τους κώδικες …wasp, για ένα ζεύγος πίεσης - θερμοκρασίας. Στο Παράρτημα της Διπλωματικής Εργασίας, δίνονται συμπληρωματικά και αποτελέσματα με βάση το κυρίως πρόγραμμα example, όπως ακριβώς αυτά λαμβάνονται στην οθόνη του Ηλεκτρονικού Υπολογιστή.

4.6 Λειτουργία κωδίκων

Στο σημείο αυτό παρουσιάζεται η δομή και η στρατηγική των κωδίκων …wasp σε βήματα τα οποία αναλύονται με λεπτομέρεια.

1. Οι κώδικες δέχονται ως είσοδο την πίεση P σε bar και τη θερμοκρασία T σε βαθμούς Κελσίου. Τα δεδομένα αυτά πρέπει να βρίσκονται μέσα στην περιοχή πιέσεων και θερμοκρασιών για τις οποίες ισχύουν οι κώδικες.

   Ο κώδικας αποδίδει στην έξοδο τα ακόλουθα μεγέθη (οι μονάδες τους δηλώνονται σε αγκύλες [.])

   P_s Πίεση κορεσμού αντίστοιχη στην θερμοκρασία T [bar]

   T_s Θερμοκρασία κορεσμού αντίστοιχη στην πίεση P [°C]

   ν Ειδικός όγκος [m^3/kg]

   u Ειδική εσωτερική ενέργεια [kJ/kg]

   ψ Ειδική ελεύθερη εσωτερική ενέργεια [kJ/kg]

   h Ειδική ενθαλπία [kJ/kg]

   s Ειδική εντροπία [kJ/(kgK)]

   c_p Ειδική θερμοχωρητικότητα υπό σταθερή πίεση [kJ/(kgK)]

   c_v Ειδική θερμοχωρητικότητα υπό σταθερό όγκο [kJ/(kgK)]

   γ Συντελεστής ισεντροπικής εκτόνωσης c_p/c_v

   a Ταχύτητα του ήχου [m/sec]

   μ Συντελεστής Joule - Thomson [K/MPa]

   k_T Ισοθερμοκρασιακή συμπιεστότητα [MPa^{-1}]

   η Δυναμική συνεκτικότητα kg/(ms)

   λ θερμική αγωγιμότητα W/(mK)

2. Η πρώτη κίνηση που κάνουν οι κώδικες είναι να διαπιστώσουν αν οι παράμετροι κλήσεως του έχουν διατυπωθεί με ορθότητα και αν έχουν πάρει αποδεκτές τιμές. Αν αυτό δεν συμβαίνει η εκτέλεση σταματάει και τυπώνονται σχετικά μηνύματα στην εκ ταυτότητας μονάδα εξόδου του χρήστη.

3. Στη συνέχεια οι κώδικες αναθέτουν στο υποπρόγραμμα checkpt να ελέγξει εάν η πίεση και η θερμοκρασία εισόδου βρίσκονται μέσα στην περιοχή για την οποία αυτοί

μπορούν να λειτουργήσουν. Αν η πίεση εισόδου έχει την τιμή 0 (ζητείται δηλαδή μια κατάσταση κορεσμού) ελέγχεται αν η θερμοκρασία εισόδου βρίσκεται στο διάστημα στο οποίο υπάρχουν καταστάσεις κορεσμού. Στην περίπτωση που ο έλεγχος είναι θετικός ανατίθεται στην function fps να υπολογίσει την πίεση κορεσμού του ελαφρού ύδατος ή του βαρέος ύδατος. Αν η θερμοκρασία εισόδου έχει την τιμή 0 (ζητείται δηλαδή και πάλι μια κατάσταση κορεσμού) ελέγχεται αν η πίεση εισόδου βρίσκεται στο διάστημα στο οποίο υπάρχουν καταστάσεις κορεσμού. Αν και εδώ είναι θετικός ο έλεγχος καλείται η function fts με σκοπό να υπολογισθεί μία αρχική εκτίμηση της θερμοκρασίας κορεσμού (TS) τον ελαφρού ή του βαρέος ύδατος σε συνάρτηση με την πίεση κορεσμού. Στην συνέχεια δίνεται εντολή στο υποπρόγραμμα newton να λύσει την εξίσωση κορεσμού $P_s = P_s(T_s)$ του ελαφρού ή του βαρέος ύδατος (function fps) ως προς την ακριβή θερμοκρασία κορεσμού χρησιμοποιώντας ως τιμή εκκίνησης για την εύρεση της λύσης την προηγούμενη αρχική εκτίμηση και βέβαια την παράγωγο της εξίσωσης $Ps = Ps(T_s)$ ως προς τη θερμοκρασία (function dfps). Με καθορισμένη την πίεση και την θερμοκρασία εξακριβώνεται πόσο κοντά είναι η κατάσταση εισόδου σε κάποια κατάσταση κορεσμού (το κριτήριο είναι πόσο κοντά είναι η θερμοκρασία εισόδου στη θερμοκρασία κορεσμού για την πίεση εισόδου -καθορίζεται ο δείκτης IS). Υπολογίζονται η ανηγμένη πίεση P_r, (μεταβλητή PR) και η ανηγμένη θερμοκρασία , T_r (μεταβλητή TR). Με τη βοήθειά τους και με τη χρησιμοποίηση των υποπρογραμμάτων function fps και function prl, που καθορίζουν τα όρια των υποπεριοχών προσδιορίζεται ακριβώς η υποπεριοχή τον διαγράμματος P-T, στην οποία βρίσκεται η πίεση και η θερμοκρασία που επιλέχθηκαν σαν είσοδος για τους κώδικες. Με την ολοκλήρωση αυτών των διεργασιών το υποπρόγραμμα checkpt έχει ενημερώσει τις μεταβλητές του script transient PP, PR (PP πίεση σε bar), TT,TR (TT θερμοκρασία σε K) και τη λίστα παραμέτρων (TS, JR, IS) και επιστρέφει τον έλεγχο στη function …wasp.

4. Η function …wasp ανάλογα με την τιμή που έχει πάρει ακριβώς ο δείκτης JR αναθέτει είτε στην function densf, είτε στην function densg, είτε και στις δύο αυτές συναρτήσεις αν πρόκειται για την κατάσταση κορεσμού, να υπολογίσουν την πυκνότητα του ελαφρού ύδατος ή του βαρέος ύδατος.

5. Στην περίπτωση που καλείται η function densf επιδιώκεται να υπολογισθεί η πυκνότητα του ελαφρού ή του βαρέος ύδατος στην υγρή φάση, σε καταστάσεις

κορεσμένου υγρού και σε ένα μέρος της υπερκρίσιμης φάσης (εκείνου που φαίνεται να είναι συνέχεια της υγρής φάσεως). Η function densf καλεί με τη σειρά της την function svlwl. Στην τελευταία έχει ανατεθεί ο υπολογισμός μιας πρώτης προσέγγισης για την πυκνότητα (DL σε g/cm^3) του ελαφρού ή του βαρέος ύδατος αποκλειστικά για τις υποπεριοχές που απασχολούν την function densf (πρόκειται για αυτές που χαρακτηρίζονται από τις τιμές 1, 4, 5 και 6 του δείκτη JR). Αναλυτικότερα, αν είναι JR = 1 ή JR = 6 υπολογίζεται η ποσότητα χ_1 (βλ. και Schmidt, 1986, σελ. 193). Αν είναι JR = 4 ή JR = 5 χρησιμοποιείται η εξίσωση που υλοποιείται στη function fpr4, η οποία εξαιτίας του ότι είναι πεπλεγμένη ως προς την πυκνότητα επιλύεται αριθμητικά. Για την αριθμητική επίλυσή της καλείται και πάλι η function newton, η οποία χρησιμοποιεί την παράγωγο της εξίσωσης που υλοποιείται από την function fpr4 ως προς την πυκνότητα (function dfpr4). Ως τιμή εκκινήσεως χρησιμοποιείται μία "ψευδοπυκνότητα" η οποία προκύπτει με την υπόθεση ότι η εξίσωση για τις υποπεριοχές όπου JR = 1 ή JR = 6 ισχύει και στις υποπεριοχές όπου JR = 4 ή JR = 5. Όταν ολοκληρωθεί ο υπολογισμός της πρώτης προσέγγισης για την πυκνότητα, ο έλεγχος επιστρέφει στην function densf και καλείται η newton για την επίλυση της εξίσωσης P = P(ρ,T) (function fp) ως προς την ακριβή τιμή της πυκνότητας του ελαφρού ή του βαρέος ύδατος. Τιμή εκκίνησης για τη λύση είναι βέβαια αυτή η αρχική προσέγγιση και χρειάζεται επιπλέον και η εξίσωση της παραγώγου της P = P(ρ,T) ως προς την πυκνότητα (function dfp). Με την εύρεση και της πυκνότητας τον ελαφρού ή του βαρέος ύδατος για την πίεση και τη θερμοκρασία εισόδου, παίρνει τιμή η μεταβλητή DF (DF η πυκνότητα σε g/cm^3) και η ροή τον κώδικα οδηγεί και πάλι στη function …wasp.

6. Στην άλλη περίπτωση που καλείται η function densg επιδιώκεται να υπολογισθεί η πυκνότητα του ελαφρού ή του βαρέος ύδατος στην ατμώδη φάση, σε καταστάσεις κορεσμένου ατμού και σε ένα μέρος της υπερκρίσιμης φάσεως (εκείνου πού φαίνεται να είναι συνέχεια της ατμώδους φάσεως). Η function densg καλεί στη συνέχεια την function svlwv. Εκεί υπολογίζεται μία πρώτη προσέγγιση για την πυκνότητα (DV σε g/cm^3) του ελαφρού ή του βαρέος ύδατος μόνο για τις υποπεριοχές που απασχολούν την function densg (πρόκειται για αυτές που χαρακτηρίζονται από τις τιμές 2, 3, 5 και 6 του δείκτη JR). Αναλυτικότερα, αν είναι JR = 2 ή JR = 6 χρησιμοποιείται η

υπολογίζεται η ποσότητα χ_2 (βλ. και Schmidt, 1986, σελ. 194). Αν είναι JR = 3 ή JR = 5 υπολογίζεται η ποσότητα β_3 (βλ. και Schmidt, 1986, σελ. 195) – function fpr3, η οποία εξαιτίας του ότι είναι και αυτή πεπλεγμένη ως προς την πυκνότητα επιλύεται αριθμητικά. Η εύρεση της λύσης ανατίθεται στη function newton. Χρησιμοποιείται η παράγωγος της εξίσωσης για την ποσότητα β_3 ως προς την πυκνότητα [function dfpr3]. Η τιμή εκκινήσεως είναι μία "ψευδοπυκνότητα", που προκύπτει αν θεωρηθεί ότι η εξίσωση για τις υποπεριοχές όπου JR= 2 ή 1R = 6 ισχύει και στις υποπεριοχές όπου JR = 3 ή JR = 5. Δεδομένου ότι υπάρχουν περιπτώσεις όπου η τιμή εκκινήσεως που υπολογίζεται με αυτόν τον τρόπο δεν είναι ικανοποιητικά "κοντά" στην πραγματική ώστε να υπολογίζεται η λύση από τη function newton, τότε χρησιμοποιείται η καταστατική εξίσωση του τέλειου αερίου για να προκύψει μια "ψευδοπυκνότητα" ως τιμή εκκινήσεως. Όταν ολοκληρωθεί ο υπολογισμός της πρώτης προσέγγισης για την πυκνότητα, ο έλεγχος επιστρέφει στην function densg και καλείται η newton για την επίλυση της εξίσωσης P = P(ρ,T) [function fp] ως προς την ακριβή τιμή της πυκνότητας τον ελαφρού ύδατος ή του βαρέος ύδατος. Τιμή εκκίνησης για τη λύση είναι βέβαια αυτή η αρχική προσέγγιση και χρειάζεται επιπλέον και η εξίσωση της παραγώγου της P = P(ρ,T) ως προς την πυκνότητα [function dfp]. Σημειώνεται ότι σε κάθε έξοδο από την function densg (μετά την κλήση της newton) ενημερώνεται η τιμή της μεταβλητής DD (= πυκνότητα σε g/cm^3) μέσω του script transient. Με την εύρεση και της πυκνότητας του ελαφρού ύδατος ή του βαρέος ύδατος για την πίεση και τη θερμοκρασία εισόδου, παίρνει τιμή η μεταβλητή DG (DG η πυκνότητα σε g/cm^3) στο script transient και η ροή τον κώδικα οδηγεί και πάλι στο υποπρόγραμμα…wasp.

7. Κάθε κλήση της function fp και της function dfp συνεπάγεται και την κλήση της function qmust για τον υπολογισμό ενός πλήθους ποσοτήτων που είναι συναρτήσεις της θερμοκρασίας και της πυκνότητας [βλέπε και την εξίσωση (3-4) και τις (4-7) έως και (4-17) στον Πετρόπουλο, 2003]. Αυτό χρησιμεύει όχι μόνο για τους υπολογισμούς που αφορούν αποκλειστικά στις function fp και dfp αλλά και για την ενημέρωση των τιμών των μεταβλητών Q, QDT, Q2D2T, QTD, Q2T2D, Q2DT, PSI0, PSI0T, PSI02T2 στο script transient, διότι χρειάζεται να χρησιμοποιούνται και σε πολλά άλλα υποπρογράμματα.

8. Το υποπρόγραμμα light_(heavy_)wasp υπολογίζει τώρα τον ειδικό όγκο του ελαφρού

(βαρέος) ύδατος SVF (υγρή φάση) ή SVG (ατμός) σε m^3/kg. Σε αυτό το σημείο εξετάζει επίσης μήπως τυχαίνει η πυκνότητα που υπολογίσθηκε από τα υποπρογράμματα densf και densg και η θερμοκρασία εισόδου (ως θερμοκρασία εισόδου εννοείται και η θερμοκρασία κορεσμού που υπολογίσθηκε από το υποπρόγραμμα checkpt για μία πίεση κορεσμού -πραγματική θερμοκρασία εισόδου 0) να βρίσκονται στη γειτονιά τον κρίσιμου σημείου. Αν πράγματι συμβαίνει αυτό η εκτέλεση σταματάει και τυπώνεται σχετικό μήνυμα στην εκ ταυτότητας μονάδα εξόδου. Αν όχι τότε καλείται η function total με σκοπό τον υπολογισμό των άλλων θερμοφυσικών ιδιοτήτων που μπορεί να δώσει ο κώδικας. Η function total καλείται μία μόνο φορά αν ο δείκτης JR παίρνει τις τιμές 1, 2, 3 και 4. Συγκεκριμένα, αν είναι JR = 1 ή JR = 4 η function total φροντίζει για τον υπολογισμό των θερμοφυσικών ιδιοτήτων τον ελαφρού ύδατος ή του βαρέος ύδατος στην υγρή φάση και σε ένα μέρος της υπερκρίσιμης φάσης. Αλλιώς, αν είναι JR = 2 ή JR = 3 η function total φροντίζει για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού ύδατος ή του βαρέος ύδατος στην ατμώδη φάση και σε ένα μέρος της υπερκρίσιμης φάσης. Στην περίπτωση που JR = 6 ή JR = 5 τότε η function total καλείται δύο φορές διότι αυτές οι τιμές του δείκτη JR αντιπροσωπεύουν καταστάσεις κορεσμού για τις οποίες οι θερμοφυσικές ιδιότητες του ελαφρού ύδατος ή του βαρέος ύδατος πρέπει να υπολογισθούν δύο φορές -μία φορά για το κορεσμένο υγρό και μία φορά για τον κορεσμένο ατμό.

9. Ανάλογα με την τιμή που έχει πάρει ο διακόπτης JP κατά την κλήση του κώδικα το υποπρόγραμμα total καλεί μία, ή δύο, ή τρεις, κτλ ή και όλες από τις ακόλουθες function: energy, enthalpy, entropy, shp, shv, shr_sove, ibm_jtc, viscosity, conductive και tension. Ο τρόπος με τον οποίο πραγματοποιούνται αυτές οι κλήσεις μέσω του διακόπτη JP εξηγήθηκε σαφώς στην §4.3. Το σύνολο αυτών των υποπρογραμμάτων (και της function total συμπεριλαμβανομένης) ανήκει στα προαιρετικά υποπρογράμματα του κώδικα, στα υποπρογράμματα δηλαδή που μπορούν να αφαιρεθούν (ένα - ένα, κατά τυχαίες ομάδες ή και όλα) χωρίς με αυτό να επηρεάζονται κατά κανένα τρόπο οι υπόλοιπες λειτουργίες τον κώδικα που περιγράφτηκαν προηγούμενα και αφορούν κυρίως στον υπολογισμό τον ειδικού όγκου του ελαφρού ύδατος για μία πίεση και μία θερμοκρασία εισόδού. Από την function energy υπολογίζεται η ειδική εσωτερική ενέργεια U και η ειδική ελεύθερη εσωτερική ενέργεια

PSI. Από την function enthalpy υπολογίζεται η ειδική ενθαλπία H. H function entropy προσδιορίζει την ειδική εντροπία S. Οι function shp και shv υπολογίζουν τις ειδικές θερμοχωρητικότητες CP και CV αντίστοιχα. H function shr_sove ασχολείται με τον ισεντροπικό εκθέτη GAMMA και την ταχύτητα του ήχου A ενώ η function ibm_jtc προσδιορίζει το συντελεστή ισοθερμοκρασιακής συμπιεστότητας IBM και το συντελεστή Joule - Thomson JTC. Η εύρεση της δυναμικής συνεκτικότητας MU και της θερμικής αγωγιμότητας K έχει ανατεθεί στις function viscosity και conductive αντίστοιχα. Το καθήκον της function tension είναι να υπολογίζει την επιφανειακή τάση sigma μεταξύ νερού και ατμού στην κατάσταση κορεσμού. Όλες οι παραπάνω function δέχονται σαν είσοδο την θερμοκρασία TT ή TR και την πυκνότητα DD με την βοήθεια του script transient (με εξαίρεση την tension η οποία δέχεται ως είσοδο μόνο τη θερμοκρασία). Για τις function που υπολογίζουν θερμοδυναμικές ιδιότητες είναι απαραίτητη και η χρησιμοποίηση κάποιων ή όλων από τις μεταβλητές Q, QDT, Q2D2T, QTD, Q2T2D, Q2DT, PSIO, PSI0T, PSI02T2, που υπολογίζονται στη function qmust. Οι μεταβλητές αυτές δεν χρειάζονται στις function viscosity, conductive και tension για τον υπολογισμό των ιδιοτήτων μεταφοράς. Με την συμπλήρωση των κλήσεων των ζητούμενων να κληθούν (διακόπτης JP) function για τον υπολογισμό των θερμοφυσικών ιδιοτήτων (όλων των function αν ζητούνται όλες – JP = 31 ή JP = 63) ενημερώνεται με τις υπολογισθείσες τιμές η λίστα παραμέτρων εξόδου

```
B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,A,MU,K,PRANDTL,SIGM
A
```

της function total και τέλος ο έλεγχος επιστρέφει στην function light_(heavy_)wasp.

10. Στο υποπρόγραμμα light_(heavy_)wasp δρομολογούνται τώρα οι τελευταίες υπολογιστικές ενέργειες, οι οποίες πρέπει να γίνουν πριν ο έλεγχος της ροής επιστρέψει στο κυρίως πρόγραμμα του χρήστη: ενημερώνονται δηλαδή αυτόματα οι τιμές των μεταβλητών που αφορούν και στο κυρίως πρόγραμμα του χρήστη με τη βοήθεια του script transient. Μετά από αυτά ο έλεγχος επιστρέφει στο κυρίως πρόγραμμα του χρήστη. Μία πολύ απλή μορφή τέτοιου προγράμματος μπορεί να είναι αυτή του κυρίως προγράμματος example. Πέρα από το script transient ο κώδικας χρησιμοποιεί και άλλο ένα script το constants. Πρόκειται για script, μέσω του οποίου

παίρνουν τιμές σταθερές, οι οποίες χρησιμοποιούνται κατά κανόνα σε περισσότερα από ένα υποπρογράμματα των κωδίκων light_(heavy_)wasp (ή γενικότερα περισσότερο από μία φορά).

Σημειώνεται ότι στον Πίνακα 4.4 παρουσιάζονται όσες από τις μεταβλητές του κώδικα δεν συμπεριλαμβάνονται στον Πίνακα 4.1. Η γνώση των μεταβλητών αυτών είναι απαραίτητη σε εκείνον το χρήστη που επιθυμεί να ασχοληθεί σε βάθος με την προγραμματιστική υλοποίηση του πλέγματος των εξισώσεων για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού ύδατος ή του βαρέος ύδατος.

4.7 Σχόλια και συμπεράσματα

Στις παραγράφους που προηγήθηκαν έγινε η παρουσίαση των αριθμητικών κωδίκων με τα ονόματα light_wasp (light water and steam properties) και heavy_wasp(heavy water and steam properties), σε επίπεδο δομής και λογικής λειτουργίας. Περιγράφηκαν διεξοδικά τα γενικά χαρακτηριστικά των κωδίκων και οι αναγκαίες οδηγίες χρήσεως. Εξετάσθηκαν όλα τα υποπρογράμματα κατά ομάδες ανάλογα και με τις λειτουργίες που αυτά διεκπεραιώνουν και παρουσιάσθηκε η ροή της εκτέλεσης ανάλογα και με τα ζητούμενα που καθορίζει ο χρήστης. Η συνολική παρουσίαση συνοδεύθηκε από συγκεντρωτικούς πίνακες πού χρησιμεύουν ως περιληπτική αναφορά στα κυριότερα σημεία των κωδίκων. Στο επόμενα $5^{ο}$ Κεφάλαιο, παρουσιάζονται οι κώδικες light_wasp και heavy_wasp αντίστοιχα με περισσότερη λεπτομέρεια, σε επίπεδο περιγραφής της εσωτερικής λειτουργίας των υποπρογραμμάτων.

ΠΙΝΑΚΕΣ ΤΟΥ 4ου ΚΕΦΑΛΑΙΟΥ

AF	ταχύτητα του ήχου a στο υγρό σε m/s (F - Fluid)
AG	ταχύτητα του ήχου a στον ατμό σε m/s (G - Gas)
CPF	ειδική θερμοχωρητικότητα c_p του υγρού υπό σταθερή πίεση σε kJ/(kgK) (Heat Capacity of Fluid at constant Pressure)
CPG	ειδική θερμοχωρητικότητα c_p του ατμού υπό σταθερή πίεση σε kJ/(kgk) (Heat Capacity of Gas at constant Pressure)
CVF	ειδική θερμοχωρητικότητα c_v του υγρού υπό σταθερό όγκο σε kJ/(kgK) (Heat Capacity of Fluid at constant Volume)
CVG	ειδική θερμοχωρητικότητα c_v του ατμού υπό σταθερό όγκο σε kJ/(kgK) (Heat Capacity of Gas at constant Volume)
GAMMAF	λόγος των ειδικών θερμοχωρητικοτήτων του υγρού CPF/CVF
GAMMAG	λόγος των ειδικών θερμοχωρητικοτήτων του ατμού CPGICVG
HF	ειδική ενθαλπία h του υγρού σε kJ/kg
HG	ειδική ενθαλπία h του ατμού σε kJ/kg
HFG	λανθάνουσα θερμότητα ατμοποίησης r σε kJ/kg (F Fluid, G = Gas)
IBMF	ισοθερμοκρασιακή συμπιεστότητα K_T του υγρού σε MPa^{-1} (Isothermal Bulk Modulus of Fluid)
IBMG	ισοθερμοκρασιακή συμπιεστότητα K_T του ατμού σε MPa^{-1} (Isothermal Bulk Modulus of Gas)
IS	δείκτης σύγκρισης με την κατάσταση κορεσμού (S = Saturation)
JP	καθορισμός των ζητουμένων ιδιοτήτων
JR	καθορισμός της περιοχής του διαγράμματος PT (R = Region)
JS	καθορισμός των μεταβλητών εισόδου
JTCF	συντελεστής Joule - Thomson μ για το υγρό σε K/MPa (Joule - Thomson Coefficient of Fluid)
JTCG	συντελεστής Joule - Thomson μ για τον ατμό σε K/MPa (Joule - Thomson Coefficient of Gas)
KF	θερμική αγωγιμότητα λ του υγρού σε W/(mK)
KG	θερμική αγωγιμότητα λ του ατμού σε W/(mK)
MUF	δυναμική συνεκτικότητα η του υγρού σε kg/(ms) ***…συνεχίζεται…***

MUG	δυναμική συνεκτικότητα η του ατμού σε kg/(ms)
P	πίεση P σε bar
PRANDTLF	αριθμός Prandtl (Pr) του υγρού
PRANDTLG	αριθμός Prandtl (Pr) του ατμού
PSIF	ειδική ελεύθερη εσωτερική ενέργεια ψ του υγρού σε kJ/kg
PSIG	ειδική ελεύθερη εσωτερική ενέργεια ψ του ατμού σε kJ/kg
SIGMAFG	επιφανειακή τάση σ μεταξύ υγρού και ατμού στην κατάσταση κορεσμού σε N/m^2
SF	ειδική εντροπία s του υγρού σε kJ/(kgK)
SG	ειδική εντροπία s του ατμού σε kJ1(kgK)
SVF	ειδικός όγκος v του υγρού σε m^3/kg (Specific Volume of Fluid)
SVG	ειδικός όγκος v του ατμού σε m^3/kg (Specific Volume of Gas)
T	θερμοκρασία T σε μονάδες K
TS	θερμοκρασία κορεσμού Ts σε μονάδες K (Saturation Temperature)
UF	ειδική εσωτερική ενέργεια u του υγρού σε kJ/kg
UG	ειδική εσωτερική ενέργεια u του ατμού σε kJ/kg

Σημείωση:

Το επίθεμα F = Fluid αναφέρεται βέβαια κυρίως στις θερμοφυσικές ιδιότητες της υγρής φάσης αλλά καλύπτει και τις ιδιότητες του κορεσμένου υγρού και ενός μέρους της υπερκρίσιμης φάσης·

Ομοίως το επίθεμα G = Gas αναφέρεται κυρίως στις θερμοφυσικές ιδιότητες της ατμώδους φάσεως αλλά καλύπτει και τις ιδιότητες του κορεσμένου ατμού και ενός διαφορετικού μέρους της υπερκρίσιμης φάσης

Πίνακας 4.1

Σύμβολα, ονοματολογία και μονάδες βασικών μεταβλητών των κωδίκων

JP = 0 δεν υπολογίζεται καμία ιδιότητα,

JP = 1 = 2^0 υπολογίζεται η εσωτερική ενέργεια,

JP = 2 = 2^1 υπολογίζονται η ενθαλπία και η εντροπία,

JP = 4 = 2^2 υπολογίζονται η ειδική θερμοχωρητικότητα υπό σταθερή πίεση, η ειδική θερμοχωρητικότητα υπό σταθερό όγκο, ο λόγος των ειδικών θερμοχωρητικοτήτων, η ταχύτητα του ήχου, η ισοθερμοκρασιακή συμπιεστότητα και ο συντελεστής Joule –Thomson

JP = 8 = 2^3 υπολογίζονται η δυναμική συνεκτικότητα, η θερμική αγωγιμότητα και ο αριθμός Prandtl,

JP = 16= 2^4 υπολογίζεται η επιφανειακή τάση, και

JP = 32 = 2^5 φυλάσσεται για μελλοντική χρήση

Ο διακόπτης IP καθορίζει ποιες ιδιότητες ζητάει ο χρήστης να υπολογισθούν από τον κώδικα. Ο διακόπτης αυτός μπορεί να πάρει τις ακέραιες τιμές από 0 έως και 63. Οι τιμές αυτές αντιπροσωπεύουν κάποιο άθροισμα των αριθμών 0, 2, 4, 8, 16 και 32 - δηλαδή ισοδύναμα των αριθμών 0, 2^0, 2^1, 2^2, 2^3, 2^4 και 2^5. Κάθε τέτοιο δυαδικό άθροισμα εξυπηρετεί τον με μοναδικό τρόπο τον προσδιορισμό του συνδυασμού των ιδιοτήτων που ζητούνται.

Πίνακας 4.2

Υπολογισμός ιδιοτήτων ελαφρού ή βαρέος ύδατος ανάλογα με τις τιμές του διακόπτη JP

IS είναι 0

όταν η πίεση ή/και η θερμοκρασία εισόδου είναι υπερκρίσιμες

IS είναι 1

όταν για αυτή την πίεση εισόδου υπάρχει κατάσταση κορεσμού. Υπολογίζεται η αντίστοιχη θερμοκρασία κορεσμού σε μονάδες K.

IS είναι 2

όταν η πίεση και η θερμοκρασία εισόδου αντιστοιχούν ακριβώς -ή βρίσκονται πολύ κοντά (θερμοκρασιακή απόσταση ±1 °C)- στην κατάσταση κορεσμού.

Πίνακας 4.3

Τιμές δείκτη IS

A	ταχύτητα του ήχου σε m/sec (function shr_sove και total)
CP	ειδική θερμοχωρητικότητα υπό σταθερή πίεση σε kJ/(kgK) (function shp και total)
CV	ειδική θερμοχωρητικότητα υπό σταθερό όγκο σε kJ/(kgK) (function shv και total)
DF	πυκνότητα του υγρού σε g/cm^3 (D = Density, F = Fluid) (function densf και light_wasp ή heavy_wasp)
DG	πυκνότητα του ατμού σε g/cm^3 (D = Density, G = Gas) (function densg και light_wasp ή heavy-wasp)
DL	πυκνότητα του υγρού σε g/cm^3 (L - Liquid) (function svlwl και densf)
DV	πυκνότητα του ατμού σε g/cm^3 (V - Vapor) (function svlwv και densg)
EPS	ειδική μεταβλητή για την function newton (παίρνει τιμές στις function svlwl, svlwv, densf, densg και checkpt)
GAMMA	λόγος ειδικών θερμοχωρητικοτήτων (function shr_sove και total)
H	ενθαλπία σε kJ/kg (function enthalpy και total)
HDT	$(\partial h/\partial \rho)_\tau$ (function ibm_jtc, shr_sove και shp)
HTD	$(\partial h/\partial T)_\rho$ (function ibm_jtc, shr_sove και shp)
Q	ποσότητα που δίνεται από τη σχέση (3-4) (βλ. και Πετρόπουλος, 2003) (function ibm_jtc, shr_sove, shp, entropy, enthalpy, energy, fp, dfpd και qmust)
QDT	$(\partial Q/\partial \rho)_\tau$ (function ibm_jtc, shr_sove, shp, enthalpy, fp, dfpd και qmust)
QTD	$(\partial Q/\partial \tau)_\rho$(function ibm_jtc, shr_sove, shp, entropy, enthalpy, energy και qmust)
Q2DT	$\partial^2 Q/(\partial \rho \partial \tau)$ (function ibm_jtc, shr_sove, shp και qmust)

…συνεχίζεται…

Q2D2T	$(\partial^2 Q/\partial^2 \rho)_T$ (function ibm_jtc, shr_sove, shp, dfpd και qmust)
Q2T2D	$(\partial^2 Q/\partial^2 \tau)_\rho$ (function ibm_jtc, shr_sove, shv και qmust)
IBM	ισοθερμοκρασιακή συμπιεστότητα σε MPa^{-1}, (υπορουτίνες ibm_jtc και total)
IER	ειδική μεταβλητή για την function newton (βλέπε και τις function svlwl, svlwv, densf, densg και checkpt)
ITMAX	ειδική μεταβλητή για την function newton (βλέπε και τις function svlwl, svlwv, densf, densg και checkpt)
JTC	συντελεστής Joule - Thomson σε K/MPa (function ibm_jtc και total)
K	θερμική αγωγιμότητα σε W/(mK) (function conductive και total)
MU	δυναμική συνεκτικότητα σε kg/(ms)
NDEC	ειδική μεταβλητή για την function newton (παίρνει τιμές στις function svlwl, svlwv, densf, densg και checkpt)
PRANDTL	αριθμός Prandtl
PC	κρίσιμη πίεση σε bar (script constants και function dfps, fps και checkpt)
PMAX	το άνω όριο της πίεσης λειτουργίας του κώδικα σε bar (script constants και function checkpt)
PMIN	το κάτω όριο της πίεσης λειτουργίας του κώδικα σε bar (script constants και function checkpt)
PDT	$(\partial P/\partial \rho)_T$ (function ibm_jtc, shr_sove και shp)
PP	πίεση σε bar [function checkpt, light_wasp (ή heavy_wasp), fp, fps και fts]
PR	ανηγμένη πίεση [function svlwl, svlwv, checkpt, light_wasp (ή heavy_wasp) και fpr3]
PRTRDR	$(\partial P_r/\partial T_r)_{\rho r}$, (function conductive)

…συνεχίζεται…

PSI	ελεύθερη εσωτερική ενέργεια σε kJ/kg (function energy και total)
PSI0	ελεύθερη εσωτερική ενέργεια αναφοράς σε kJ/kg (function energy και enthalpy)
PSI0P	$d\psi_0/dT$ (function entropy, enthalpy, energy και qmust)
PSI02T2	$d^2\psi_0/d^2T$ (function ibm_jtc, shr_sove, shv, shv και qmust)
PTD	$(\partial P/\partial T)_\rho$ (function ibm_jtc, shr_sove και shp)
S	εντροπία σε kJ/(kgk) (function entropy και total)
SIGMA	επιφανειακή τάση σε N/m^2 (function tension και total)
TC	κρίσιμη θερμοκρασία σε K [script constants, function checkpt και light_wasp (ή heavy_wasp, prs, fps και dfps]
TMAX	το άνω θερμοκρασιακό όριο λειτουργίας του κώδικα σε K (script constants και function checkpt)
TMIN	το κάτω θερμοκρασιακό όριο λειτουργίας του κώδικα σε K (script constants και function checkpt)
TR	ανηγμένη θερμοκρασία [function svlwl, svlwv, checkpt, light_wasp (ή heavy_wasp), fpr4, dfpr4, fpr3, dfpr3, prl και prs]
TT	θερμοκρασία σε K (function viscosity, ibm_jtc, shr_sove, shv, shp, entropy, enthalpy, energy και qmust)
U	εσωτερική ενέργεια σε kJ/kg (function energy και total)
VC	κρίσιμος ειδικός όγκος (script constants και function svlwl και svlwv)
XTR	$\rho_r(\partial\rho_r/\partial P_r)_{Tr}$ (function viscosity και conductive)
X1	ανηγμένος ειδικός όγκος στις υποπεριοχές 1 και 6 (function svlwl)
X2	ανηγμένος ειδικός όγκος στις υποπεριοχές 2 και 6 (function svlwv)

…συνεχίζεται…

X3 ανηγμένος ειδικός όγκος στις υποπεριοχές 3 και 5 (function fpr3, dfpr3 και svlwv)

X4 ανηγμένος ειδικός όγκος στις υποπεριοχές 4 και 5 (function fpr4, dfpr4 και svlwl)

Πίνακας 4.4

Σύμβολα, ονοματολογία και μονάδες μεταβλητών του κώδικα

ΚΕΦΑΛΑΙΟ 5
ΚΩΔΙΚΕΣ ΥΠΟΛΟΓΙΣΜΟΥ ΘΕΡΜΟΦΥΣΙΚΩΝ ΙΔΙΟΤΗΤΩΝ ΕΛΑΦΡΟΥ ΚΑΙ ΒΑΡΕΟΣ ΥΔΑΤΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ MATLAB & SCILAB

5.1 Εισαγωγή

Παρουσιάζονται στη συνέχεια ταυτόχρονα όλα τα υποπρογράμματα των κωδίκων light_wasp και heavy_wasp στο περιβάλλον προγραμματισμού MATLAB (έκδοση R2009a) και SCILAB (έκδοση 5.2.2) δεδομένου ότι οι δύο κώδικες μοιράζονται σημαντικά κοινά μέρη και ότι τα δύο περιβάλλοντα προγραμματισμού είναι παρόμοια. Τα διαφορετικά σημεία των κωδίκων στις διαφορές που αφορούν στους υπολογισμούς για το ελαφρύ και το βαρύ νερό αποδίδονται με γραμμοσκιασμένα μέρη. Ιδιαίτερη έμφαση δίνεται στην εξέταση των υποπρογραμμάτων light_wasp (ή heavy_wasp), checkpt, qmust και total, τα οποία είναι τα μεγαλύτερα και σημαντικότερα υποπρογράμματα. Τα υπόλοιπα υποπρογράμματα παρουσιάζονται συνοπτικότερα. Το σύνολο των υποπρογραμμάτων παρουσιάζεται κατά το δυνατόν με τη σειρά που χρησιμοποιούνται στον κώδικα. Οι κυριότερες συνιστώσες της παρουσίασης κάθε υποπρογράμματος είναι:

α. Μεταβλητές εισόδου

β. Μεταβλητές εξόδου

γ. Τοπικές μεταβλητές

δ. Υποπρογράμματα που καλούν το παρουσιαζόμενο υποπρόγραμμα και

ε. Υποπρογράμματα που καλεί το παρουσιαζόμενο υποπρόγραμμα

Σημειώνεται ότι δεν είναι σκόπιμη -και ως εκ τούτου δε γίνεται- η εξέταση του υποδείγματος κυρίως προγράμματος εφαρμογής example. Ο χρήστης του κώδικα που θα θελήσει να πειραματισθεί με αυτό μπορεί πολύ εύκολα να το χρησιμοποιήσει, έχοντας βέβαια πάντοτε υπόψη τις οδηγίες που δίνονται στο 4^{ο} Κεφάλαιο, σχετικά με τη σύνδεση ενός κυρίου προγράμματος με τον κώδικα light_wasp (ή heavy_wasp). Από την άλλη κρίθηκε σκόπιμο, να παρουσιασθούν και τα scripts initial, constants και transient. Κατά

την εξέταση κάθε υποπρογράμματος, αν τύχει να υπάρχει μεταβλητή εισόδου ή εξόδου, η οποία χρειάζεται να επικοινωνεί με τα scripts αυτό σημειώνεται με κατάλληλο τρόπο.

Ειδικά σε αυτό το Κεφάλαιο, η αρίθμηση των εξισώσεων στις οποίες γίνεται αναφορά είναι αυτή των εξισώσεων της Διδακτορικής Διατριβής του Πετρόπουλου (2003), εκτός αν σημειώνεται διαφορετικά.

5.2 Λεπτομερής παρουσίαση του κώδικα

5.2.1 Το υποπρόγραμμα light_wasp (ή heavy_wasp)

Μεταβλητές εισόδου (κατά σειρά εμφανίσεως)

στη λίστα παραμέτρων

JS (πρέπει να έχει πάντα την τιμή 1)

JP (πρέπει να έχει πάντα ακέραια τιμή από 0 έως 63)

P (πίεση σε bar μέσα στην περιοχή ισχύος του κώδικα ή 0)

T (θερμοκρασία σε K μέσα στην περιοχή ισχύος του κώδικα ή 0)

από scripts

TC (κρίσιμη θερμοκρασία σε K -από το script constants.m)

PP (πίεση σε bar από script transient.m)

TT (θερμοκρασία σε K απο script transient.m)

PSEUDODC (ψευδοκρίσιμη πυκνότητα σε g/cm^3)

στις λίστες παραμέτρων καλούμενων υποπρογραμμάτων

DF (πυκνότητα του υγρού σε g/cm^3)

DG (πυκνότητα του ατμού σε g/cm^3)

Μεταβλητές εξόδου (κατά σειρά εμφανίσεως)

στη λίστα παραμέτρων

P (πίεση σε bar, αν η P εισόδου ήταν 0 και υπάρχει κατάσταση κορεσμού)

T (θερμοκρασία σε °C, αν η T εισόδου ήταν 0 και υπάρχει κατάσταση κορεσμού)

TS (θερμοκρασία κορεσμού σε °C, αν υπάρχει κατάσταση κορεσμού)

SVF (ειδικός όγκος του υγρού σε m^3/kg)

SVG (ειδικός όγκος του ατμού σε m^3/(kg)

JR (υποπεριοχή του διαγράμματος PT, ακέραιες τιμές από 1 έως 6)

IS (ακέραιες τιμές από 0 έως 2)

στο script transient

PP (πίεση σε bar)

PR (ανηγμένη πίεση)

TT (θερμοκρασία σε K)

TR (ανηγμένη θερμοκρασία)

JJ (=JR, υποπεριοχή του PT)

UF, PSIF, HF, SF, CPF, CVF, IBMF, JTCF, GAMMAF, AF, MUF, KF, PRANDTLF

UG, PSIG, HG, SG, CPG, CVG, IBM C, JTCG, GAMMAG, AG, MUG, KG, PRANDTLG

HFG, SIGMAFG, LACFG

Δεν υπάρχουν τοπικές μεταβλητές.

Το light_wasp (heavy_wasp) καλείται από:

Το κυρίως πρόγραμμα του χρήστη.

Το light_wasp (heavy_wasp) καλεί τα υποπρογράμματα του κώδικα:

checkpt, densf, densg, total

Το light_wasp (heavy_wasp) **είναι απαραίτητο υποπρόγραμμα του κώδικα.**

(βλέπε τα κείμενα των υποπρογραμμάτων light_wasp και heavy_wasp στο Παράρτημα)

5.2.2 Το υποπρόγραμμα checkpt

Μεταβλητές εισόδου (κατά σειρά εμφανίσεως)

στο script constants.m

PMIN (ελάχιστη πίεση σε bar για την οποία ισχύει ο κώδικας)

PMAX (μέγιστη πίεση σε bar για την οποία ισχύει ο κώδικας)

TMIN (ελάχιστη θερμοκρασία σε K για την οποία ισχύει ο κώδικας)

TMAX (μέγιστη θερμοκρασία σε K για την οποία ισχύει ο κώδικας)

PC (κρίσιμη πίεση σε bar)

TC (κρίσιμη θερμοκρασία σε K)

TRT (ανηγμένη θερμοκρασία στο κρίσιμο σημείο)

TR1, TR2, TR3, PR1, PR2

(όρια για τις θερμοκρασίες και τις πιέσεις που καθορίζουν τις υποπεριοχές -από το script constants.m)

στο script transient.m

PP (πίεση σε bar)

TT (θερμοκρασία σε K)

στις λίστες παραμέτρων καλούμενων υποπρογραυμάτων

IER (δείκτης λάθους που συνέβη στο υποπρόγραμμα newton)

Μεταβλητές εξόδου (κατά σειρά εμφανίσεως)

στη λίστα παραμέτρων

T (θερμοκρασία κορεσμού σε K, αν υπάρχει κατάσταση κορεσμού)

JR (υποπεριοχή τον διαγράμματος PT, JR = 1 έως 6)

IS (δείκτης του πόσο κοντά είναι η είσοδος σε μία κατάσταση κορεσμού

IS = 0 υπερκρίσιμη πίεση και θερμοκρασία εισόδου

IS = 1 υποκρίσιμη πίεση και θερμοκρασία εισόδου

IS = 2 η κατάσταση κορεσμού απέχει μόλις 1 K)

στο script transient.m

PP (πίεση σε bar)

PR (ανηγμένη πίεση)

TT (θερμοκρασία σε K)

TR (ανηγμένη θερμοκρασία)

Τοπικές Μεταβλητές

EPS, NDEC, ITMAX

(χρήσιμες για την κλήση του υποπρογράμματος newton)

Το checkpt καλείται από:

Το υποπρόγραμμα light_wasp (ή το heavy_wasp)

Το υποπρόγραμμα checkpt καλεί τα υποπρογράμματα τον κώδικα:

fps, fts, dfps, newton και prl

Το checkpt είναι **απαραίτητο υποπρόγραμμα του κώδικα.**

(βλέπε τα κείμενα του υποπρογράμματος checkpt στο Παράρτημα)

5.2.3 Το υποπρόγραμμα qmust

Μεταβλητές εισόδου (κατά σειρά εμφανίσεως)

απο scripts

R [σταθερή του ελαφρού (βαρέος) ύδατος - από το script constants.m]

TT (θερμοκρασία σε K - από το script transient.m)

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

Μεταβλητές εξόδου (κατά σειρά εμφανίσεως)

στο script transient.m

Q, QDT, Q2D2T, QTD, Q2T2D, Q2DT, PSI0, PSI0T, PSI02T2

Πρόκειται για ποσότητες που δίνονται από την εξίσωση (3-4) και τις εξισώσεις (4-7) έως και (4-17) στον Πετρόπουλο, 2003.

Τοπικές Μεταβλητές

A (πίνακας σταθερών Aij για τις εξισώσεις)

C (πίνακας σταθερών Ci για τις εξισώσεις)

TAJ1, TAJ2**,** E, RAJl, RAJ2

(άλλες σταθερές των εξισώσεων -βλέπε και Keyes F.G. et al., 1968)

I, J (δείκτες για επαναληπτικές διαδικασίες)

SUMIm (m-οστό άθροισμα SUM ως προς το δείκτη I, m = 1 έως 9)

SUMJn (n-οστό άθροισμα SUM ως προς το δείκτη J, n = l έως 6)

Η qmust καλείται από:

Τα υπαπρογράμματα fp και dfp

Η qmust δεν καλεί κανένα υποπρόγραμμα.

Η qmust είναι **απαραίτητο υποπρόγραμμα του κώδικα.**

(βλέπε τα κείμενα του υποπρογράμματος qmust στο Παράρτημα)

5.2.4 Το υποπρόγραμμα total

Μεταβλητές εισόδου (κατά σειρά εμφανίσεως)

στη λίστα παραμέτρων

JP (διακόπτης που καθορίζει ποιές από τις θερμοφυσικές ιδιότητες θα υπολογισθούν, JP = 1 έως 63)

U, PSI, H, S. CP, CV, IBM, JTC, GAMMA,A, MU, K, PRANDTL, SIGMA

(στην είσοδο όλες οι παραπάνω μεταβλητές έχουν μηδενική τιμή, πράγμα που επιτυγχάνεται με τη χρήση του script initial.m)

στα script

JJ (υποπεριοχή του διαγράμματος PT, JJ = 1 έως 6 - από το script transient.m)

JPC1, JPC2, JPC3, JPC4

(πίνακες από το script constants που περιέχουν σταθερές, οι οποίες διευκολύνουν τους χειρισμούς που πρέπει να γίνουν από την total για να ικανοποιηθεί το αίτημα του χρήστη που υποβλήθηκε με τον διακόπτη JP)

Μεταβλητές εξόδου (κατά σειρά εμφανίσεως)

στι λίστα παραμέτρων

U, PSI, H, S, CP, CV, IBM, 7TC, GAMMA, A, MU, K, PRANDTL, SIGMA

Τοπικές Μεταβλητές

I (δείκτης για επαναληπτικές διαδικασίες)

Το total καλείται από:

Το υποπρόγραμμα light_wasp (ή το heavy_wasp)

Το total καλεί τα υποπρογράμματα του κώδικα:

energy, enthalpy, entropy, shp, shv, shr_sove, newton, ibm_itc, viscosity, conductive, tension

(Το πόσα από αυτά τα υποπρογράμματα θα κληθούν ή όχι εξαρτάται από την τιμή του δείκτη JP).

Το total είναι **προαιρετικό υποπρόγραμμα του κώδικα.**

(βλέπε τα κείμενα του υποπρογράμματος total στο Παράρτημα)

5.2.5 Τα υπόλοιπα υποπρογράμματα

5.2.5.1 Το υποπρόγραμμα fts

Πρόκειται για υλοποίηση της εξίσωσης (4-32), ή (5-2) για το βαρύ νερό, (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

PP (πίεση σε bar - από το script transient.m)

A [πίνακας σταθερών της εξίσωσης (4-32), ή (5-2) για το βαρύ νερό - από το script constants.m]

Μεταβλητές εξόδου:

FTS (θερμοκρασία σε K)

Τοπικές μεταβλητές: ΔΕΝ ΥΠΑΡΧΟΥΝ

(το υποπρόγραμμα καλείται από το υποπρόγραμμα checkpt και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος fts στο Παράρτημα)

5.2.5.2 Το υποπρόγραμμα fps

Πρόκειται για -υλοποίηση της εξίσωσης (3-23), ή (3-43) για το βαρύ νερό (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

TS (θερμοκρασία κορεσμού σε K - από το script transient.m)

PC (κρίσιμη πίεση σε bar - από το script constants.m)

TC (κρίσιμη θερμοκρασία σε K - από το script constants.m)

K [πίνακας σταθερών για την εξίσωση (3-23) - από το script constants.m]

ή A1, A2, A4, A11, A20

[σταθερές για την εξίσωση (3-43) – από το script constants.m]

PP (πίεση σε bar - από το script transient.m)

Μεταβλητές εξόδου:

FPS (πίεση σε MPa)

Τοπικές μεταβλητές:

PPC (κρίσιμη πίεση σε MPa)

PPP (πίεση σε MPa)

TAUS (ειδική μορφή της ανηγμένης θερμοκρασίας κορεσμού)

(το υποπρόγραμμα καλείται από το υποπρόγραμμα newton -στο υποπρόγραμμα checkpt – και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος fps στο Παράρτημα)

5.2.5.3 Το υποπρόγραμμα dfps

Πρόκειται για υλοποίηση της εξίσωσης (4-31), ή (5-1) για το βαρύ νερό (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

TS (θερμοκρασία κορεσμού σε K - από το script transient.m)

PC (κρίσιμη πίεση σε bar - από το script constants.m)

TC (κρίσιμη θερμοκρασία σε K - από το script constants.m)

K [πίνακας σταθερών για την εξίσωση (3-23) - από το script constants.m]

ή A1, A2, A4, A11, A20

[σταθερές για την εξίσωση (3-43) – από το script constants.m]

Μεταβλητές εξόδου:

Η παράγωγος DFPS

Τοπικές μεταβλητές:

FPSS [πίεση κορεσμού σε MPa -σχέση (3-23)]

TAUS (ειδική μορφή της ανηγμένης θερμοκρασίας κορεσμού)

(το υποπρόγραμμα καλείται από το υποπρόγραμμα newton -στο υποπρόγραμμα checkpt– και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος dfps στο Παράρτημα)

5.2.5.4 Το υποπρόγραμμα densf

Μεταβλητές εισόδου:

DL [πυκνότητα τον ελαφρού (βαρέος) ύδατος σε g/cm^3 από το υποπρόγραμμα svlwl]

IER (δείκτης λάθους που μπορεί να συμβεί στο υποπρόγραμμα newton)

Μεταβλητές εξόδου: DF, DD (πυκνότητα σε g/cm^3)

Τοπικές μεταβλητές: EPS, NDEC, ITMAX (χρήσιμες για την κλήση του υποπρογράμματος newton)

[το υποπρόγραμμα καλείται από το light_wasp (ή το heavy_wasp) και καλεί τα υποπρογράμματα svlwl, fp, dfpd και newton]

(βλέπε τα κείμενα του υποπρογράμματος densf στο Παράρτημα)

5.2.5.5 Το υποπρόγραμμα densg

Μεταβλητές εισόδου:

DV [πυκνότητα του ελαφρού (βαρέος) ύδατος σε g/cm^3 από το υποπρόγραμμα svlwv]

IER (δείκτης λάθους που μπορεί να συμβεί στο υποπρόγραμμα solve)

Μεταβλητές εξόδου: DG, DD (πυκνότητα σε g/cm^3)

Τοπικές μεταβλητές: EPS, NDEC, ITMAX (χρήσιμες για την κλήση του υποπρογράμματος newton)

[το υποπρόγραμμα καλείται από το light_wasp (ή το heavy_wasp) και καλεί τα υποπρογράμματα svlwv, fp, dfpd και newton]

(βλέπε το κείμενο του υποπρογράμματος densg στο 6^{ο} Κεφάλαιο)

5.2.5.6 Το υποπρόγραμμα svlwl

Πρόκειται για υλοποίηση των εξισώσεων (4-18) και (4-26), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

PC, VC (κρίσιμη πίεση και κρίσιμος ειδικός όγκος - από το script constants.m)

R [σταθερή του ελαφρού (βαρέος) ύδατος - από το script constants.m]

PR (ανηγμένη πίεση - από το script transient.m)

TR (ανηγμένη θερμοκρασία - από το script transient.m)

JJ (=JR, υποπεριοχή - από το script transient.m)

IER (δείκτης λάθους που συνέβη στο υποπρόγραμμα newton - από το script transient.m)

Μεταβλητές εξόδου: PR (ανηγμένη πίεση), DL (αρχική προσέγγιση για την πυκνότητα του ελαφρού (βαρέος) ύδατος σε g/cm^3)

Τοπικές μεταβλητές:

AM [πίνακας σταθερών a_i, i = 1 έως 12 -εξίσωση (4-18)]

A [πίνακας σταθερών A_i, i = 11 έως 22 -εξίσωση (4-18)]

EPS, NDEC, ITMAX

(χρήσιμες για την κλήση του υποπρογράμματος newton)

Y [ποσότητα που δίνεται από την εξίσωση (4-20)]

Z [ποσότητα που δίνεται από την εξίσωση (4-19)]

X1 [ανηγμένος ειδικός όγκος του ελαφρού ύδατος στις υποπεριοχές 1 και 6 – εξίσωση (4-18)]

X4 [ανηγμένος ειδικός όγκος του ελαφρού ύδατος στις υποπεριοχές 4 και 5 - εξίσωση 4-26)]

VL (ειδικός όγκος του ελαφρού ύδατος σε cm^3/g)

(το υποπρόγραμμα καλείται από το densf και καλεί το prs καθώς και το newton)

(βλέπε τα κείμενα του υποπρογράμματος svlwl στο Παράρτημα)

5.2.5.7 Το υποπρόγραμμα svlwv

Πρόκειται για υλοποίηση των εξισώσεων (4-21) και (4-25), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

PC, VC (κρίσιμη πίεση και κρίσιμος ειδικός όγκος - από το script constants.m)

R (σταθερή του ελαφρού ύδατος - από το script constants.m)

PR (ανηγμένη πίεση - από το script transient.m)

TR (ανηγμένη θερμοκρασία - από το script transient.m)

JJ (=JR, υποπεριοχή - από το script transient.m)

IER (δείκτης λάθους που συνέβη στο υποπρόγραμμα newton)

Μεταβλητές εξόδου: PR (ανηγμένη πίεση), DV (αρχική προσέγγιση για την πυκνότητα του ελαφρού (βαρέος) ύδατος σε g/cm^3)

Τοπικές μεταβλήτες:

B0, B90 [σταθερές για την εξίσωση (4-21)]

B [πίνακας σταθερών, i = 1 έως 9, j = 1 έως 6 -εξίσωση (4-21) ομοίως]

NM [πίνακας σταθερών n(μ)]

ZM [πίνακας σταθερών z(μ,ν)]

LM [πίνακας σταθερών l(μ)]

XM [πίνακας σταθερών x(μ,λ)]

EPS, NDEC, ITMAX

(χρήσιμες για την κλήση του υποπρογραμματος newton)

I1 [παράμετρος της εξίσωσης (4-21)-σχέση (4-22)]

X [παράμετρος της εξίσωσης (4-21) -σχέση (4-23)]

VSM1, VSM2, VSN, VSN1, VSN2, VSL

(μερικά αθροίσματα ως προς μ, ν και λ)

X2 [ανηγμένος ειδικός όγκος του ελαφρού ύδατος στις υποπεριοχές 2 και 6 - εξίσωση (4-21)]

X3 [ανηγμένος ειδικός όγκος του ελαφρού ύδατος στις υποπεριοχές 3 και 5 - εξίσωση (4-25)]

VV (ειδικός όγκος τον ελαφρού ύδατος σε cm^3/g)

(το υποπρόγραμμα καλείται από το densg και καλεί τα fps και prl καθώς και το υποπρόγραμμα newton)

(βλέπε τα κείμενα του υποπρογράμματος svlwv στο Παράρτημα)

5.2.5.8 Το υποπρόγραμμα prl

Πρόκειται για υλοποίηση της εξίσωσης (4-24), (βλ. και Πετρόπουλος 2003).

Μεταβλητές εισόδου:

TR (ανηγμένη θερμοκρασία - από το script transient.m)

TR1, TR2 [ανηγμένες θερμοκρασίες - από το script constants.m –σταθερές για την εξίσωση (4-24)]

PR1, PR2 [ανηγμένες πιέσεις - από το script constants.m -σταθερές για την εξίσωση (4-24)]

Μεταβλητές εξόδου: PRL (β_L στον Schmidt, 1987)

Τοπικές μεταβλητές: L [σταθερή για την εξίσωση (4-24)]

(το υποπρόγραμμα καλείται από τα υποπρόγραμματα checkpt, svlwl και svlwv και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος prl στο Παράρτημα)

5.2.5.9 Το υποπρόγραμμα fpr3

Πρόκειται για υλοποίηση της εξίσωσης (4-25), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

X3

[κακή προσέγγιση του ανηγμένου ειδικού όγκου του ελαφρού ύδατος στις υποπεριοχές 3 και 5 με βάση την εξίσωση (4-21). Χρησιμοποιείται για την εύρεση καλύτερης προσέγγισης από την εξίσωση (4-25). Η καλύτερη αυτή προσέγγιση τον ανηγμένου ειδικού όγκου είναι απαραίτητη για τη λύση της εξίσωσης (3-2) της παρούσας Διπλωματικής Εργασίας, ως προς την ακριβή τιμή της πυκνότητας για τις υποπεριοχές 3 και 5 αν είναι βεβαίως δεδομένες η πίεση και η θερμοκρασία (βλέπε και τα υποπρόγραμματα densg, svlwv και newton)].

C(64) [πίνακας σταθερών για την εξίσωση (4-25) - από το script constants.m - δεν χρησιμοποιούνται όλες]

C010, 0011, C012, C310

[άλλες σταθερές για την εξίσωση (4-25) - από το script constants.m]

PR (ανηγμένη πίεση - από το script transient.m)

TR (ανηγμένη θερμοκρασία - από το script transient.m)

Μεταβλητές εξόδου: FPR3 (β_3 στον Schmidt, 1987)

Τοπικές μεταβλητές:

XSC0N, XSC1N, XSC2N, XSC3N, XSC4N, XSC6N

[δηλαδή γενικά μεταβλητές της μορφής XSCi,n με i = 0, 1, 2, 3, 4 και 6 και n από 0 έως και 9, οι οποίες συμβολίζουν απαραίτητα μερικά αθροίσματα (S = SUM) της εξίσωσης (4-25) για τον υπολογισμό του ανηγμένου ειδικού όγκου X που σχηματίζονται κυρίως χρησιμοποιώντας τις σταθερές C(in), όπου i είναι δεκάδες και n είναι μονάδες]

N (δείκτης για επαναληπτικές διαδικασίες)

(το υποπρόγραμμα καλείται από το newton -στο svlwv- καθώς και από το fpr4 και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος fpr3 στο Παράρτημα)

5.2.5.10 Το υποπρόγραμμα dfpr3

Πρόκειται για υλοποίηση της εξίσωσης (4-29), (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

X3 (βλέπε σχετικά στην περιγραφή του fpr3)

C(64) [πίνακας σταθερών για την εξίσωση (4-29) - από το script constants.m δεν χρησιμοποιούνται όλες]

C010, 0011, C012, C310

[άλλες σταθερές για την εξίσωση (4-29) - από το script constants.m]

TT (θερμοκρασία σε K -δεν χρησιμοποιείται - από το script transient.m)

TR (ανηγμένη θερμοκρασία - από το script transient.m)

Μεταβλητές εξόδου: DFPR3

Τοπικές μεταβλητές:

DXSC0N, DXSC1N, DXSC2N, DXSC3N, DXSC4N, DXSC6N

[δηλαδή γενικά μεταβλητές της μορφής DXSCi,n με i = 0, 1, 2, 3, 4 και 6 και n από 0 έως και 9, οι οποίες συμβολίζουν απαραίτητα μερικά αθροίσματα (S = SUM) της εξίσωσης (4-29) για τον υπολογισμό της παραγώγου (D = Derivative) της σχέσεως (4-25) ως προς τον ανηγμένο ειδικό όγκο X, που σχηματίζονται κυρίως χρησιμοποιώντας τις σταθερές C(in), όπου i είναι δεκάδες και n είναι μονάδες]

N (δείκτης για επαναληπτικές διαδικασίες)

(το υποπρόγραμμα καλείται από το newton –στο svlwv - καθώς και από το dfpr4 και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος dfpr3 στο Παράρτημα)

5.2.5.11 Το υποπρόγραμμα fpr4

Πρόκειται υλοποίηση της εξίσωσης (4-26), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

X4

[κακή προσέγγιση του ανηγμένου ειδικού όγκου του ελαφρού ύδατος στις υποπεριοχές 4 και 5 με βάση την εξίσωση (4-18). Χρησιμοποιείται για την εύρεση καλύτερης προσέγγισης από την εξίσωση (4-26). Η καλύτερη αυτή προσέγγιση του ανηγμένού ειδικού όγκου είναι απαραίτητη για τη λύση της εξίσωσης (3-2) στην παρούσα Διπλωματική Εργασία ως προς την ακριβή τιμή της πυκνότητας για τις υποπεριοχές 4 και 5 αν είναι βεβαίως δεδομένες η πίεση και η θερμοκρασία (βλέπε και τα υποπρογράμματα densf, svlwl και newton)].

TRl [ανηγμένη θερμοκρασία - από το script constants.m -σταθερή για την εξίσωση (4-26)]

D(5,4) [πίνακας σταθερών για την εξίσωση (4-26) - από το script constants.m - δεν χρησιμοποιούνται όλες)

D30, D40, D50

[άλλες σταθερές για την εξίσωση (4-26) - από το script constants.m]

TR (ανηγμένη θερμοκρασία - από το script transient.m)

Μεταβλητές εξόδου: FPR4 (β_4 στον Schmidt, 1987)

Τοπικές μεταβλητές:

Y [σχέση (4-27)]

XSDN1, XSDN2

[μερικά αθροίσματα απαραίτητα για τον προσδιορισμό του ανηγμένου ειδικού όγκου στις υποπεριοχές 4 και 5 με βάση την εξίσωση (4-26)]

XSDMN

[μεταβλητή της μορφής XSDm,n με m = 3 ή 4 και n από I έως και 4. Είναι και αυτή απαραίτητο μερικό άθροισμα της εξίσωσης (4-27) για τον υπολογισμό του ανηγμένου ειδικού όγκου, το οποίο σχηματίζεται κυρίως χρησιμοποιώντας τις σταθερές D(m,n)]

M, N (δείκτες για επαναληπτικές διαδικασίες)

(το υποπρόγραμμα καλείται από το newton -στο svlwl και καλεί το fpr3)

(βλέπε τα κείμενα του υποπρογράμματος fpr3 στο Παράρτημα)

5.2.5.12 Το υποπρόγραμμα dfpr4

Πρόκειται για υλοποίηση της εξίσωσης (4-30), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

X4 (βλέπε σχετικά στην περιγραφή του υποπρόγραμμα fpr4)

TR1 [ανηγμένη θερμοκρασία - από το script constants.m -σταθερή για την εξίσωση (4-27)]

D(5,4) [πίνακας σταθερών για την εξίσωση (4-30) - από το script constants.m - δεν χρησιμοποιούνται όλες]

D30, D40, D50

[άλλες σταθερές για την εξίσωση (4-30) - από το script constants.m]

TR (ανηγμένη θερμοκρασία - από το script constants.m)

Μεταβλητές εξόδου: DFPR4

Τοπικές μεταβλήτες:

Y [σχέση (4-29)]

DXSDNI, DXSDN2

[μερικά αθροίσματα απαραίτητα για του προσδιορισμό της παραγώγου της εξίσωσης (4-28) ως προς τον ανηγμένο ειδικό όγκο στις υποπεριοχές 4 και 5]

DXSDMN

[μεταβλητή της μορφής DXSDm,n με m = 3 ή 4 και n από 1 έως και 4. Είναι και αυτή απαραίτητο μερικό άθροισα για τον υπολογισμό της παραγώγου της εξίσωσης (4-28) ως προς του ανηγμένο ειδικά άγκο, το οποίο σχηματίζεται κυρίως χρησιμοποιώντας τις σταθερές D(m,n)]

M, N (δείκτες για επαναληπτικές διαδικασίες)

(το υποπρόγραμμα καλείται από το newton -στο υποπρόγραμμα svlwl, και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος dfpr4 στο Παράρτημα)

5.2.5.13 Το υποπρόγραμμα fp

Πρόκειται για υλοποίηση της εξίσωσης (3-7), (βλ. και Πετρόπουλος 2003). Η εξίσωση αυτή είναι ανάλογη στην ουσία η (3-2) της παρούσας Διπλωματικής Εργασίας.

Μεταβλητές εισόδου:

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

R (σταθερή του ελαφρού ή βαρέος ύδατος - από το script constants.m)

PP (πίεση σε bar - από το script transient.m)

T (θερμοκρασία σε K - από το script transient.m)

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8) –υποπρόγραμμα qmust, - από το script transient.m]

Q [ποσότητα που υπολογίζεται από τη σχέση (4-8) –υποπρόγραμμα qmust, - από το script transient.m]

Μεταβλητέ εξόδου: FP (πίεση σε MPa)

Τοπικές μεταβλητές: PPP (πίεση σε MPa)

(το υποπρόγραμμα καλείται από το υποπρόγραμμα newton -στα υποπρογράμματα densf και densg - και καλεί το υποπρόγραμμα qmust)

(βλέπε τα κείμενα του υποπρογράμματος fp στο Παράρτημα)

1.4.5.14 Το υποπρόγραμμα dfpd

Πρόκειται για υλοποίηση της εξίσωσης (4-1), (βλ. και Πετρόπουλος 2003).]

Μεταβλητές εισόδου:

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

T (θερμοκρασία σε K - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4), υποπρόγραμμα qmust - από το script transient.m]

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8), υποπρόγραμμα qmust - από το script transient.m]

Q2D2T [ποσότητα που υπολογίζεται από τη σχέση (4-10), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: ΔΕΝ ΥΠΑΡΧΟΥΝ

Τοπικές μεταβλητές: ΔΕΝ ΥΠΑΡΧΟΥΝ

(το υποπρόγραμμα καλείται από το υποπρόγραμμα newton -στα υποπρόγραμματα densf και densg- και καλεί το υποπρόγραμμα qmust)

(βλέπε τα κείμενα του υποπρογράμματος qmust στο Παράρτημα)

5.2.5.15 Το υποπρόγραμμα newton

Πρόκειται για υλοποίηση της αριθμητικής μεθόδου Newton – Raphson (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

f	[συνάρτηση f = f(x) = 0, πεπλεγμένη ως προς x για την οποία ζητείται ρίζα(-ες) x]
df	[συνάρτηση για την παράγωγο df(x)/dx]
EPS	[κριτήριο σύγκλισης, μία ρίζα x γίνεται αποδεκτή αν είναι |f(x)|≤EPS]
NDEC	(κριτήριο σύγκλισης, μία ρίζα x γίνεται αποδεκτή αν δύο διαδοχικές προσεγγίσεις της συμφωνούν στα NDEC πρώτα δεκαδικά τους ψηφία)
X	(αρχική προσέγγιση της ζητούμενης ρίζας)
ITMAX	(μέγιστος επιτρεπόμενος αριθμός επαναλήψεων)

(Σημείωση: Στις περισσότερες από τις περιπτώσεις υπολογισμών που έγιναν με το υποπρόγραμμα light_wasp ή heavy_wasp η ζητούμενη ρίζα x βρέθηκε σε λιγότερες από 40 επαναλήψεις. Εξαιρείται συνήθως η γειτονιά του κρίσιμου σημείου και ορισμένες περιοχές στην υγρή φάση. Οπως όμως και να έχει το πράγμα το υποπρόγραμμα newton περιορίζει εσωτερικά τον μέγιστο επιτρεπόμενο αριθμό επαναλήψεων στις 100. Αν δηλαδή ο χρήστης έχει ζητήσει κατά την κλήση του υποπρογράμματος να είναι ITMAX > 100, τα αίτημά του αυτό αγνοείται. Συνήθεις τιμές για τις μεταβλητές εισόδου EPS, NDEC και ITMAX είναι οε $I0^{-5}$, 5 και 100 αντίστοιχα.)

Μεταβλητές εξόδου:

X	(ζητούμενη ρίζα x)
IER	(προειδοποιητική παράμετρος λάθους)

(Σημείωση: Αν επιτευχθεί η εύρεση της ζητούμενης ρίζας x τότε επιστρέφεται IER = 0. Αν η ζητούμενη ρίζα x δεν είναι δυνατόν να βρεθεί επειδή η παράγωγος df παίρνει κάποια στιγμή τιμή που είναι "πολύ κοντά" στο 0, τότε επιστρέφεται IER = 1 και X = 111111. Αν η ζητούμενη ρίζα x δεν είναι δυνατόν να βρεθεί επειδή χρειάζονται για αυτό περισσότερες

από ITMAX επαναλήψεις, τότε επιστρέφεται IER = 2 και X= 222222. Αν η ζητούμενη ρίζα x δεν είναι δυνατόν να βρεθεί επειδή συμβαίνουν και τα δύο αυτά ατυχή γεγονότα, τότε επιστρέφεται IER = 3 και X = 111111 ή X = 222222.)

Τοπικές μεταβλητές:

K (δείκτης για την επαναληπτική διαδικασία)

TOL (κριτήριο σύγκλισης)

(Σημείωση: Σύγκλιση επιτυγχάνεται όταν $|x_k - x_{k-1}| \leq TOL$*. Η τιμή της TOL ε:ίναι* 10^{-NDEC} *για τις επαναλήψεις από 1 έως 40,* $10^{(-NDEC-1)}$ *για τις επαναλήψεις από 41 έως 60,* $10^{(-NDEC-2)}$ *για τις επαναλήψεις από 61 έως 80 και* $10^{(-NDEC-3)}$ *για τις επαναλήψεις από 81 έως 100. Στις λίγες περιπτώσεις όπου οι επαναλήψεις ξεπερνούν τις 40 - γειτονιά του κρισίμου σημείου κτλ- η ζητούμενη ρίζα x που τελικά υπολογίζεται με την αυξημένη ανοχή TOL είναι η καλύτερη πον μπορεί να ληφθεί από τις κάθε φορά συναρτήσεις προς επίλυση.)*

Το υποπρόγραμμα newton καλείται από τα υποπρογράμματα checkpt, densf, densg, svlwl, svlwv και δεν καλεί κανένα υποπρόγραμμα. Αναλυτικότερα:

- καλείται απο τπο υποπρόγραμμα checkpt με σκοπό την επίλυση της εξισώσεως κορεσμού [υποπρόγραμμα fps -σχέση (3-23) για το ελαφρύ νερό ή (3-43) για το βαρύ νερό] ως προς τη θερμοκρασία κορεσμού αν το μόνο δεδομένο είναι η πίεση κορεσμού. Η προσέγγιση της λύσης δίνεται από το υποπρόγραμμα fts -σχέση (4-32) για το ελαφρύ νερό ή σχέση (5-2) για το βαρύ νερό- ενώ η παράγωγος της συνάρτησης FPS υπολογίζεται στο υποπρόγραμμα dfps -σχέση (4-31) για το ελαφρύ νερό ή σχέση (5-1) για το βαρύ νερό.
- καλείται από το υποπρόγραμμα densf με σκοπό την επίλυση της εξισώσεως (3-3), υποπρόγραμμα fp, ως προς την πυκνότητα για τις υποπεριοχές 1, 4, 5 και 6, αν τα δεδομένα είναι η πίεση και η θερμοκρασία. Η προσέγγιση της λύσης δίνεται από το υποπρόγραμμα svlwl -σχέσεις (4-18) και (4-26)- ενώ η παράγωγος της συνάρτησης FP ως προς την πυκνότητα υπολογίζεται στο υποπρόγραμμα dfpd -σχέση (4-1).
- καλείται από το υποπρόγραμμα densg με σκοπό την επίλυση της εξισώσεως (3-3), υποπρόγραμμα fp, ως προς την πυκνότητα για τις υποπεριοχές 1, 3, 5 και 6, αν

τα δεδομένα είναι η πίεση και η θερμοκρασία. Η προσέγγιση της λύσης δίνεται από το υποπρόγραμμα svlwv -σχέσεις (4-21) και (4-25)- ενώ η παράγωγος τηςσυνάρτησης FP ως προς την πυκνότητα υπολογίζεται στο υποπρόγραμμα dfpd -σχέση (4-1).

- καλείται από το υποπρόγραμμα svlwl με σκοπό την επίλυση της εξισώσεως (4-26), υποπρόγραμμα fpr4, ως προς την προσέγγιση της λύσης για την πυκνότητα στις υποπεριοχές 4 και 5, αν τα δεδομένα είναι η πίεση και η θερμοκρασία. Και εδώ χρειάζεται μία προσέγγιση για την προσέγγιση της λύσης, η οποία δίνεται από την εξίσωση (4-18) -αυ αυτή υποτεθεί ότι ισχύει και για τις υποπεριοχές 4 και 5 (κανονικά ισχύει μόνο για τις υποπεριοχές 1 και ό). Η παράγωγος της συνάρτησης FPR4 ως προς την πυκνότητα υπολογίζεται στο υποπρόγραμμα dfpr4 -σχέση (4-30).
- τέλος καλείται από το υποπρόγραμμα svlwv με σκοπό την επίλυση της εξισώσεως (4-25), υποπρόγραμμα fpr3, ως προς την προσέγγιση της λύσης για την πυκνότητα στις υποπεριοχές 3 και 5, αν τα δεδομένα είναι πίεση και η θερμοκρασία. Και εδώ χρειάζεται μία προσέγγιση για την προσέγγιση της λύσης, η οποία δίνεται είτε από την εξίσωση (4-21) -αν αυτή υποτεθεί ότι ισχύει και για τις υποπεριοχές 3 και 5 (κανονικά ισχύει μόνο για τις υποπεριοχές 2 και 6), είτε από την καταστατική εξίσωση τελείου αερίου. Η παράγωγος της συνάρτησης FPR3 ως προς την πυκνότητα υπολογίζεται στο υποπρόγραμμα dfpr3 -σχέση (4-29).

(βλέπε τα κείμενα του υποπρογράμματος newton στο Παράρτημα)

5.2.5.16 Το υποπρόγραμμα energy

Πρόκειται για υλοποίηση των εξισώσεων (3-1) και (3-9), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

R (σταθερή τον ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4), υπορουτίνα qmust - από το script transient.m]

QTD [ποσότητα που υπολογίζεται από τη σχέση (4-7), υποπρόγραμμα qmust - από το script transient.m]

PSI0 [ποσότητα που υπολογίζεται από τη σχέση (3-5), υποπρόγραμμα qmust - από το script transient.m]

PSI0T [ποσότητα που υπολογίζεται από τη σχέση (4-12), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: U (ειδική εσωτερική ενέργεια σε kJ/kg), PSI (ειδική ελεύθερη εσωτερική ενέργεια σε kJ/kg)

Τοπικές μεταβλητές: ΔΕΝ ΥΠΑΡΧΟΥΝ

(υποπρογραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος energy στο Παράρτημα)

5.2.5.17 Το υποπρόγραμμα enthalpy

Πρόκειται για υλοποίηση της εξίσωσης (3-13), (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4) για το ελαφρύ νερό ή την (3-41) για το βαρύ νερό, υποπρόγραμμα qmust - από το script transient.m]

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8), υποπρόγραμμα qmust - από το script transient.m]

QTD [ποσότητα που υπολογίζεται από τη σχέση (4-7), υποπρόγραμμα qmust - από το script transient.m]

PSI0 [ποσότητα που υπολογίζεται από τη σχέση (3-5) για το ελαφρύ νερό ή την

(3-42) για το βαρύ νερό, υπορουτίνα qmust, - από το script transient.m]

PSI0T [ποσότητα που υπολογίζεται από τη σχέση (4-12), υπορουτίνα qmust - από το script transient.m]

Μεταβλητές εξόδου: H (ειδική ενθαλπία σε kJ/kg)

Τοπικές μεταβλητές: ΔΕΝ ΥΠΑΡΧΟΥΝ

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος enthalpy στο Παράρτημα)

5.2.5.18 Το υποπρόγραμμα entropy

Πρόκειται για την υλοποίηση της εξίσωσης (3-11), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4) ή την (3-41) για το βαρύ νερό, υποπρόγραμμα qmust - από το script transient.m]

QTD [ποσότητα που υπολογίζεται από τη σχέση (4-7), υποπρόγραμμα qmust - από το script transient.m]

PSI0T [ποσότητα που υπολογίζεται από τη σχέση (4-12), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: S [ειδική εντροπία σε kJ/(kgK)]

Τοπικές μεταβλητές:ΔΕΝ ΥΠΑΡΧΟΥΝ

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος entropy στο Παράρτημα)

5.2.5.19 Το υποπρόγραμμα shp

Πρόκειται για υλοποίηση της εξίσωσης (3-15), χρησιμοποιούνται επιπλέον και οι εξισώσεις (4-1) έως και (4-5), (βλ. και Πετρόπουλος, 2003).

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4) ή την (3-41) για το βαρύ νερό, υποπρόγραμμα qmust, - από το script transient.m]

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8), υποπρόγραμμα qmust - από το script transient.m]

QTD [ποσότητα που υπολογίζεται από τη σχέση (4-7), υποπρόγραμμα qmust - από το script transient.m]

Q2T2D [ποσότητα που υπολογίζεται από τη σχέση (4-11), υποπρόγραμμα qmust - από το script transient.m]

Q2DT [ποσότητα που υπολογίζεται από τη σχέση (4-9), υποπρόγραμμα qmust - από το script transient.m]

PSI02T2 [ποσότητα που υπολογίζεται από τη σχέση (4-17), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: CP [ειδική θερμοχωρητικότητα υπό σταθερη πίεση σε kJ/(kgK)]

Τοπικές μεταβλητές:

HTD [παράγωγος $(\partial h/\partial T)_\rho$ -σχέση (4-4)]

HDT [παράγωγος $(\partial h/\partial \rho)_T$ -σχέση (4-5)]

PTD [παράγωγος $(\partial P/\partial T)_\rho$ -σχέση (4-3)]

PDT [παράγωγος $(\partial P/\partial \rho)_T$ -σχέση (4-1)]

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος shp στο Παράρτημα)

5.2.5.20 Το υποπρόγραμμα shv

Πρόκειται για υλοποίηση της εξίσωσης (3-17), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

DD (πυκνότητα σε g/cm^3 - από το script transient.m)

Q2T2D [ποσότητα που υπολογίζεται από τη σχέση (4-11), υποπρόγραμμα qmust - από το script transient.m]

PSI02T2 [ποσότητα που υπολογίζεται από τη σχέση (4-17), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: CV [ειδική θερμοχωρητικότητα υπό σταθερό όγκο σε kJ/(kgK)]
Τοπικές μεταβλητές: ΔΕΝ ΥΠΑΡΧΟΥΝ

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος shv στο Παράρτημα)

5.2.5.21 Το υποπρόγραμμα shr_sove

Πρόκειται για υλοποίηση των εξισώσεων (3-18) και (3-22), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4) ή την (3-41) για το βαρύ νερό, υποπρόγραμμα qmust - από το script transient.m]

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8), υποπρόγραμμα qmust - από το script transient.m]

Q2D2T [ποσότητα που υπολογίζεται από τη σχέση (4-10), υποπρόγραμμα qmust - από το script transient.m]

QTD [ποσότητα παν υπολογίζεται από τη σχέση (4-7), υποπρόγραμμα qmust -

από το script transient.m]

Q2T2D [ποσότητα παν υπολογίζεται από τη σχέση (4-11), υποπρόγραμμα qmust - από το script transient.m]

Q2DT [ποσότητα παν υπολογίζεται από τη σχέση (4-9), υποπρόγραμμα qmust - από το script transient.m]

PSI02T2 [ποσότητα που υπολογίζεται από τη σχέση (4-17,) υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: GAMMA (λόγος των ειδικών θερμοχωρητικοτήτων –εκθέτης ισεντροπικής μεταβολής), A (ταχύτητα τον ήχου σε m/sec)

Τοπικές μεταβλητές:

HTD [παράγωγος $(\partial h/\partial T)_\rho$ -σχέση (4-4)]

HDT [παράγωγος $(\partial h/\partial \rho)_T$ -σχέση (4-5)]

PTD [παράγωγος $(\partial P/\partial T)_\rho$ -σχέση (4-3)]

PDT [παράγωγος $(\partial P/\partial \rho)_T$ -σχέση (4-1)]

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος shr_sove στο Παράρτημα)

5.2.5.22 Το υποπρόγραμμα ibm_jtc

Πρόκειται για υλοποίηση των εξισώσεων (3-20) και (3 21), (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

R (σταθερή του ελαφρού ή του βαρέος ύδατος - από το script constants.m)

TT (θερμοκρασία σε K - από το script transient.m)

Q [ποσότητα που υπολογίζεται από τη σχέση (3-4), υποπρόγραμμα qmust - από το script transient.m]

QDT [ποσότητα που υπολογίζεται από τη σχέση (4-8), υποπρόγραμμα qmust - από το script transient.m]

Q2D2T [ποσότητα που υπολογίζεται από τη σχέση (4-10), υποπρόγραμμα qmust - από το script transient.m]

QTD [ποσότητα που υπολογίζεται από τη σχέση (4-7) υποπρόγραμμα qmust, - από το script transient.m]

O2T2D [ποσότητα που υπολογίζεται από τη σχέση (4-11) υποπρόγραμμα qmust, - από το script transient.m]

Q2DT [ποσότητα που υπολογίζεται από τη σχέση (4-9), υποπρόγραμμα qmust - από το script transient.m]

PSI02T2 [ποσότητα πού υπολογίζεται από τη σχέση (4-17), υποπρόγραμμα qmust - από το script transient.m]

Μεταβλητές εξόδου: IBM (ισοθερμοκρασιακή συμπιεστότητα σε MPa^{-1}), JTC (συντελεστής Joule - Thomson σε K/MPa)

Τοπικές μεταβλητές:

HTD [παράγωγος $(\partial h/\partial T)_\rho$ -σχέση (4-4)]

HDT [παράγωγος $(\partial h/\partial \rho)_T$ -σχέση (4-5)]

PTD [παράγωγος $(\partial P/\partial T)_\rho$ -σχέση (4-3)]

PDT [παράγωγος $(\partial P/\partial \rho)_T$ -σχέση (4-1)]

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος ibm_jtc στο Παράρτημα)

5.2.5.23 Το υποπρόγραμμα viscosity

Πρόκειται για υλοποίηση της εξίσωσης (3-32) ή (3-50) αν πρόκειται για το βαρύ νερό, (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου:

για την περίπτωση του ελαφρού ύδατος

PSEUDODC (ψευδοκρίσιμη πυκνότητα - από το script constants.m), PSEUDOTC (ψευδοκρίσιμη θερμοκρασία - από το script constants.m), TT (θερμοκρασία σε K), DD (πυκνότητα σε g/cm^3),

Μεταβλητές εξόδου: MU [δυναμική συνεκτικότητα σε kg/(ms)]

Τοπικές μεταβλητές:

H	(πίνακας με τις σταθερές H_i, i = 1 έως 4)
HH	(πίνακας με τις σταθερές H_{jk}, j = 1 έως 6, k = 1 έως 7)
NC	[δυναμική συνεκτικότητα στο κρίσιμο σημείο kg/(ms)]
PSEUDODR	(ψευδοανηγμένη πυκνότητα)
PSEUDOTR	(ψευδοανηγμένη θερμοκρασία)
NR0	[υλοποίηση σχέσης (3-33)]
NR1	[υλοποίηση σχέσης (3-34)]
I, J, K	(δείκτες για επαναληπτικές διαδικασίες)
NR2	[παράγων της εξίσωσης (3-32)]
NR	(ανηγμένη δυναμική συνεκτικότητα)

για την περίπτωση του βαρέος ύδατος

PSEUDODC (ψευδοκρίσιμη πυκνότητα - από το script constants.m), PSEUDOTC (ψευδοκρίσιμη θερμοκρασία - από το script constants.m), TR (ανηγμένη θερμοκρασία), DD (πυκνότητα σε g/cm^3),

Μεταβλητές εξόδου: MU [δυναμική συνεκτικότητα σε kg/(ms)]

Τοπικές μεταβλητές:

A	(πίνακας με τις σταθερές A_{ij}, i = 1 έως 6, j = 1 έως 7)
B	(πίνακας με τις σταθερές B_k, k = 1 έως 4)
HETA	[παράγων της εξίσωσης (3-51)]
PSEUDODR	(ψευδοανηγμένη πυκνότητα)

I, J, K (δείκτες για επαναληπτικές διαδικασίες)

N0 [υλοποίηση της εξίσωσης (3-51)]

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος viscosity στο Παράρτημα)

5.2.5.24 Το υποπρόγραμμα conductive

Πρόκειται για υλοποίηση της εξίσωσης (3-35) ή (3-52) αν πρόκειται για το βαρύ νερό, (βλ. και Πετρόπουλος 2003).

για την περίπτωση του ελαφρού ύδατος

Μεταβλητές εισόδου: PSEUDODC (ψευδοκρίστμη πυκνότητα), PSEUDOTC (ψευδοκρίσιμη θερμοκρασία - - από το script constants.m), TT (θερμοκρασία σε K), DD (πυκνότητα σε g/cm^3)

Μεταβλητές εξόδου: K [Θερμική αγωγιμότητα σε W/(mK)]

Τοπικές μεταβλητές:

H (πίνακας με τις σταθερές H_i, i = 1 έως 4)

HH (πίνακας με τις σταθερές H_{jk}, j = 1 έως 6, k = 1 έως 7)

L (πίνακας με τις σταθερές L_i, i = 1 έως 4)

LL (πίνακας με τις σταθερές L_{jk}, j = 1 έως 6, k = 1 έως 7)

ELC [θερμική αγωγιμότητα στο κρίσιμο σημείο kg/(ms)]

PSEUDODR (ψευδοανηγμένη πυκνότητα)

PSEUDOTR (ψευδοανηγμένη θερμοκρασία)

NR0 [υλοποίηση σχέσης (3-33)]

NR1 [υλοποίηση σχέσης (3-34)]

ELR0 [υλοποίηση σχέσης (3-36)]

ELR1 [υλοποίηση σχέσης (3-37)]

ELR2 [υλοποίηση σχέσης (3-38)]

A, B [άλλες σταθερές για την εξίσωση (3-35)]

I, J, K (δείκτες για επαναληπτικές διαδικασίες)

για την περίπτωση του βαρέος ύδατος

Μεταβλητές εισόδου: PSEUDODC (ψευδοκρίσιμη πυκνότητα), PSEUDOTC (ψευδοκρίσιμη θερμοκρασία - από το script constants.m), TR (ανηγμένη θερμοκρασία), DD (πυκνότητα σε g/cm^3)

Μεταβλητές εξόδου: K [θερμική αγωγιμότητα σε W/(mK)]

Τοπικές μεταβλητές:

A (πίνακας με τις σταθερές A_i, i = 1 έως 5)

B (πίνακας με τις σταθερές B_j, j = 1 έως 4)

A0, B0, BE, CT1, CR1, CR2, CR3, PSEUDODR1, C1, C2, D1, LAMDA

[άλλες σταθερές για την εξίσωση (3-52)]

PSEUDODR (ψευδοανηγμένη πυκνότητα)

EL0 [μερικό άθροισμα για την εξίσωση (3-52) –σχέση (3-53)]

DELTAEL [ομοίως -σχέση (3-54)]

ELR0 [υλοποίση σχέσης (3-36)]

F1 [συνάρτηση που χρησιμοποιείται στην εξίσωση (3-52) –σχέση (3-53)]

F2 [ομοίως - σχέση (3-58)]

F3 [ομοίως - σχέση (3-59)]

F4 [ομοίως - σχέση (3-60)]

TAUR (μία ειδική μορφή της ανηγμένης θερμοκρασίας)

DELTAELC [μερικό άθροισμα για την εξίσωση (3-52) –σχέση (3-55)]

DELTAELL [μερικό άθροισμα για την εξίσωση (3-52) –σχέση (3-56)]

I, J (δείκτες για επαναληπτικές διαδικασίες)

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος conductive στο Παράρτημα)

5.2.5.25 Το υποπρόγραμμα tension.

Πρόκειται για υλοποίηση της εξίσωσης (3-39) ή (3-61) αν πρόκειται για το βαρύ νερό, (βλ. και Πετρόπουλος, 2003)

Μεταβλητές εισόδου: TT (θερμοκρασία σε K - από το script transient.m), J(=JR, υποπεριοχή)

Μεταβλητές εξόδου: SIGMA (επιφανειακή τάση σε N/m)

Τοπικές μεταβλητές:

SIGMA0	[σταθερή για την εξίσωση (3-39) ή (3-61) αν πρόκειται για το βαρύ νερό]
M	(ομοίως)
B	(ομοίως)
TCSTR	(κρίσιμη θερμοκρασία του ελαφρού ύδατος ή του βαρέος ύδατοςσύμφωνα με τους Staub J.et al., 1980)
TAUT	(ειδική μορφή της ανηγμένης θερμοκρασίας)

(το υποπρόγραμμα καλείται από το total και δεν καλεί κανένα υποπρόγραμμα)

(βλέπε τα κείμενα του υποπρογράμματος tension στο Παράρτημα)

5.3 Σχόλια και συμπεράσματα

Παρουσιάσθηκαν στο Κεφάλαιο αυτό, όλα τα υποπρογράμματα των κωδίκων light_wasp και heavy_wasp στο περιβάλλον προγραμματισμού MATLAB (έκδοση R2009a) και SCILAB (έκδοση 5.2.1) ως συνέχεια του 4^{ου} Κεφαλαίου στο οποίο περιγράφτηκε η λογική των κωδίκων αυτών. Οι συνιστώσες της παρουσίασης κάθε υποπρογράμματος ήταν κυρίως: (α) η εξίσωση ή οι εξισώσεις οι οποίες υλοποιούνται (β) οι μεταβλητές εισόδου, (γ) οι μεταβλητές εξόδου, (δ) οι τοπικές μεταβλητές, (ε) οι μονάδες που έχουν ορισμένες από τις μεταβλητές, (στ) τα υποπρογράμματα που καλούν κάθε εξεταζόμενο υποπρόγραμμα και (ζ) τα υποπρογράμματα που καλεί το παρουσιαζόμενο υποπρόγραμμα

Ως συνέχεια του 4^{ου} και του 5^{ου} Κεφαλαίου στο Παράρτημα παρουσιάζονται τα κείμενα των υποπρογραμμάτων των κωδίκων. Στο επόμενο 6^{ο} Κεφάλαιο γίνονται αναγκαίες

παρατηρήσεις σχετικά με τις διάφορες ιδιαιτερότητες του προγραμματισμού σε MATLAB και SCILAB σε σχέση και με το περιβάλλον προγραμματισμού FORTRAN.

ΚΕΦΑΛΑΙΟ 6
ΑΝΤΙΚΑΤΑΣΤΑΣΗ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΩΝ ΕΡΓΑΛΕΙΩΝ FORTRAN ΑΠΟ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΑ ΕΡΓΑΛΕΙΑ MATLAB & SCILAB

6.1 Εισαγωγή

Σε αυτό το Κεφάλαιο, παρουσιάζεται ο τρόπος με τον οποίο αντικαταστάθηκαν ορισμένα από τα προγραμματιστικά εργαλεία που υπάρχουν στη γλώσσα FORTRAN από αντίστοιχα προγραμματιστικά εργαλεία που διατίθενται στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB. Η παρουσίαση αυτή είναι απαραίτητη διότι ορισμένες από τις εντολές και τα εργαλεία της FORTRAN υλοποιούνται κάπως ή πολύ διαφορετικά στο MATLAB ή το SCILAB. Ειδικά, σημαντικές διαφορές υπάρχουν στην έννοια του κυρίως προγράμματος, στον τύπο και τη λειτουργία των διαθέσιμων υποπρογραμμάτων, στην μεταβίβαση του ελέγχου μέσω GOTO, στον τρόπο που καθορίζονται και αλλάζουν τιμές οι τοπικές και οι καθολικές μεταβλητές, στον τρόπο που μπορούν να προσδιορίζονται οι ρίζες εξίσωσης, στον τρόπο που λειτουργούν οι εντολές stop, return, end ή ισοδύναμες, στον τρόπο που παράγεται εκτελέσιμος κώδικας και τέλος στις διαδικασίες debugging.

6.2 Κυρίως πρόγραμμα

Ένα κυρίως πρόγραμμα MATLAB ή SCILAB αποτελείται από μία αλληλουχία εντολών οι οποίες εκτελούνται διαδοχικά. Οι εντολές συγκροτούν ένα κείμενο εντολών (script) και τοποθετούνται σε ένα αρχείο κειμένου. Σε αντίθεση με την FORTRAN δεν υπάρχουν χαρακτηριστικές εντολές που σηματοδοτούν την έναρξη και τη λήξη ενός κυρίως προγράμματος. Το κυρίως πρόγραμμα μπορεί να καλεί υποπρογράμματα. Σε αντίθεση με την FORTRAN, τα υποπρογράμματα δεν μπορεί να είναι μέρος του αρχείου κειμένου που περιέχει το script, πρέπει να βρίσκονται σε διαφορετικά αρχεία. Ολες οι εντολές MATLAB ή SCILAB συντάσσονται με "πεζούς" χαρακτήρες.

6.3 Υποπρογράμματα

Όπως έχει ήδη επισημανθεί στο 4^{ο} Κεφάλαιο, τόσο στο περιβάλλον προγραμματισμού MATLAB, όσο και σε αυτό του SCILAB διατίθενται μία και μόνο μορφή υποπρογράμματος, αυτό της συνάρτησης (function), σε αντίθεση με τη FORTRAN ή άλλες γλώσσες που διαθέτουν περισσότερες από μία μορφές. Παραδείγματος χάριν η FORTRAN διαθέτει τα υποπρογράμματα subroutine, function και blockdata. Στην περίπτωση που χρειάζεται να υπάρχει ένα κείμενο εντολών (script), οι οποίες πρέπει να επαναλαμβάνονται από καιρό σε καιρό σε διάφορα σημεία ενός κώδικα MATLAB ή SCILAB, αυτό μπορεί να τοποθετείται σε χωριστό αρχείο του κώδικα και να καλείται / εκτελείται όταν χρειάζεται. Όλα τα ονόματα των υποπρογραμμάτων πρέπει να είναι σε "πεζά". Υπάρχουν χαρακτηριστικές εντολές που σηματοδοτούν την έναρξη και τη λήξη ενός υποπρογράμματος function, τον τρόπο που αυτό συναλλάσσει τιμές μεταβλητών με τα υπόλοιπα μέρη ενός κώδικα MATLAB ή SCILAB και τον τρόπο που αυτό επιστρέφει τον έλεγχο σε άλλο μέρος του κώδικα. Σε αντίθεση με τα κείμενα εντολών (script) τα υποπρογράμματα function μπορούν να περιέχουν άλλα υποπρογράμματα function. Η κλήση της function foo στο MATLAB ή στο SCILAB γίνεται όπως παρακάτω:

[out1, out2, ..., outN] = foo(in1, in2, ..., inN);

όπου outN οι μεταβλητές εξόδου και inN οι μεταβλητές εισόδου.

Από MATLAB εννοείται η κλήση χωρίς μεταβλητές εισόδου ως εξής:

[out1, out2, ..., outN] = foo;

Από SCILAB η κλήση χωρίς μεταβλητές εισόδου όμως είναι:

[out1, out2, ..., outN] = foo();

πράγμα που δεν είναι καθόλου προφανές από τα σχετικά εγχειρίδια χρήσης.

Από MATLAB και SCILAB εννοείται η κλήση χωρίς μεταβλητές εξόδου ως εξής:

[] = foo(in1, in2, ..., inN);

6.4 Εντολές GOTO

Στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, για λόγους διαφύλαξης των απαιτήσεων δομημένου προγραμματισμού, δεν διατίθεται η εντολή GOTO. Αυτό το γεγονός από μόνο του δεν είναι απαραίτητα κακό, αλλά δυστυχώς, υπάρχουν κώδικες δομημένου προγραμματισμού, στους οποίους η εντολή GOTO χρησιμοποιείται για τον έλεγχο πολλαπλών επιλογών μεταπήδησης ελεγχόμενων με πολύ αυστηρό τρόπο. Μία τέτοια περίπτωση είναι και η υπορουτίνα TOTAL, η οποία γράφθηκε σε FORTRAN και χρησιμοποιήθηκε στους κώδικες υπολογισμού των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος στον Πετρόπουλο (2003). Ο τρόπος που λειτουργεί η υπορουτίνα αυτή παρουσιάσθηκε με λεπτομέρεια σε αντίστοιχο κώδικα για τις θερμοφυσικές ιδιότητες του ελαφρού ύδατος από τον Hendricks (1973). Το κείμενο της υπορουτίνας TOTAL σε γλώσσα FORTRAN δίνεται πιο κάτω. Σημειώνεται ότι οι αναγκαίες τιμές των μεταβλητών JPC1, JPC2, JPC3 και JPC4, λαμβάνονται από υποπρόγραμμα BLOCKDATA, το οποίο επίσης δίνεται:

```
      SUBROUTINE TOTAL
     1      (JP,B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,
     2      CP,CV,IBM,JTC,GAMMA,A,MU,K,PRANDTL,SIGMA)
C
C     MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
C     IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
C     MUST BE CALLED TWICE.
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DOUBLE PRECISION IBM,JTC,K,MU
      DIMENSION JPC1(32),JPC2(32),JPC3(32),JPC4(32)
      COMMON
     1  /K4/J
     2  /Z12/JPC1,JPC2,JPC3,JPC4
C
C     ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
C     SPECIFICATION,JP
C
      IF (MOD(JP,2)) 60,70,60
C
C     VIRIAL COEFFICIENTS,1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
C     TEMPERATURE AND INTERNAL ENERGY
C
   60 CALL VIRIAL(B2,B3,B4,DB2,DB3,DB4)
      CALL ENERGY(U,PSI)
   70   DO 80 I=1,32
        IF (JP-JPC1(I)) 100,90,80
   80   CONTINUE
      GO TO 100
C
C     ENTHALPY AND ENTROPY
C
   90   CALL ENTHALPY(H)
        CALL ENTROPY(S)
  100   DO 110 I=1,32
        IF (JP-JPC2(I)) 130,120,110
  110   CONTINUE
      GO TO 130
C
C     SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT
C     VOLUME,SPECIFIC HEAT RATIO,SONIC VELOCITY,ISOTHERMAL BULK
C     MODULUS AND JOULE-THOMSON COEFFICIENT
C
  120   CALL SHP(CP)
      CALL SHV(CV)
      CALL SHR_SOVE(GAMMA,A)
      CALL IBM_JTC(IBM,JTC)
  130   DO 140 I=1,32
        IF (JP-JPC3(I)) 160,150,140
  140   CONTINUE
      GO TO 160
C
C     VISCOSITY,THERMAL CONDUCTIVITY AND PRANDTL NUMBER
C
  150   CALL VISCOSITY(MU)
      CALL CONDUCTIVE(K)
```

```
      IF (K.NE.0.D+0) PRANDTL=1000.*MU*CP/K
  160   DO 170 I=1,32
        IF (JP-JPC4(I)) 190,180,170
  170   CONTINUE
      GO TO 190
C
C     SURFACE TENSION
C
  180   IF ((J.EQ.5).OR.(J.EQ.6)) CALL TENSION(SIGMA)
  190   IF (JP-32) 200,210,210
C
C       RESERVED FOR FUTURE USE
C
  210   GO TO 200
  200   RETURN
      END
```

```
      BLOCKDATA CONSTANTS
C     BLOCKDATA SUBPROGRAM CONTAINING ALL CONSTANTS OF HEAVY_WASP
C     BEING USED AT LEAST TWICE.ALL OTHER ONCE USED CONSTANTS ARE
C     PART OF CORRESPONDING DATA COMMANDS IN INDIVIDUAL SUBPROGRAMS.
C     COMMUNICATE WITH REST OF PACKAGE USING ONLY Z LABELLED COMMON
C     COMMANDS
C
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      DOUBLE PRECISION K(9)
      DIMENSION C(64),D(5,5),JPC1(32),JPC2(32),JPC3(32),JPC4(32)
C
C     COMMONS LABELLED WITH Z REPRESENT ALL CONSTANTS OF HEAVY_WASP
C     BEING USED AT LEAST TWICE
C
      COMMON
     *  /Z1/PMIN,PMAX
     *  /Z2/TMIN,TMAX
     *  /Z3/PC,TC,VC
     *  /Z4/TRT,TR1,TR2,TR3
     *  /Z5/PR1,PR2
     *  /Z6/R
     *  /Z7/A1,A2,A4,A11,A20
     *  /Z8/K
     *  /Z9/C,C010,C011,C012,C310
     *  /Z10/D
     *  /Z11/PSEUDODC
     *  /Z12/JPC1,JPC2,JPC3,JPC4
      DATA
C
C     PRESSURE AND TEMPERATURE RANGE OF EQUATIONS FOR THERMODYNAMIC
C     AND TRANPORT PROPERTIES OF HEAVY WATER
C
     *  PMIN,PMAX/0.006601D+0,1000.D+0/,
     *  TMIN,TMAX/276.95D+0,775.D+0/,
C
C     CRITICAL PRESSURE,TEMPERATURE AND SPECIFIC VOLUME OF HEAVY WATER
C
     *  PC,TC,VC/216.6D+0,643.89D+0,2.793D+0/,
C
C     REDUCED TEMPERATURE RANGES OF THERMODYNAMIC SUB-REGIONS
C
     *  TRT,TR1,TR2,TR3
     +  /4.30120051D-1,9.626911787D-1,1.333462073D+0,1.657886606D+0/,
C
C     MAXIMUM REDUCED PRESSURE
C
     *      PR1,PR2/7.454108957D-1,4.616805171D+0/,
C
C     HEAVY WATER CONSTANT
C
     *      R/0.415147D+0/,
C
C     CONSTANTS OF HEAVY WATER SATURATION LINE
C
     *      A1,A2,A4,A11,A20
     +  /-7.81583D+0,17.6012D+0,-18.1747D+0,-3.92488D+0,4.19174D+0/
C
```

```
C     NUMERICAL VALUES OF PRIMARY CONSTANTS OF EQUATION OF STATE
C     OF LIGHT WATER BEING USED IN THIS PACKAGE
C
      DATA
C
C     SATURATION LINE
C
     *  K
     1  /-7.691234564D+0,-2.608023696D+1,-1.681706546D+2,6.423285504D+1,
     2  -1.189646225D+2,4.167117320D+0,2.097506760D+1,1.D+9,6.D+0/,
C
C     THERMODYNAMIC SUB-REGIONS 3 AND 4
C
     *  C
     1  /-1.72260420D-2,-7.77175039D+0,4.20460752D+0,-2.76807038D+0,
     2  2.10419707D+0,-1.14649588D+0,2.23138085D-1,1.16250363D-1,
     3  -8.20900544D-2,0.D+0,7.08636085D-1,1.23679455D+1,-1.20389004D+1,
     4  5.40437422D+0,-9.93865043D-1,6.27523182D-2,-7.74743016D+0,
     5  3*0.D+0,-4.29885092D+0,4.31430538D+1,-1.41619313D+1,
     6  4.04172459D+0,1.55546326D+0,-1.66568935D+0,3.24881158D-1,
     7  2.93655325D+1,2*0.D+0,7.94841842D-6,8.0885947D+1,-8.36153380D+1,
     8  3.58636517D+1,7.51895954D+0,-1.26160640D+1,1.09717462D+0,
     9  2.12145492D+0,-5.46529566D-1,2.75971776D-6,-5.09073985D-4,
     A  8*0.D+0,2.10636332D+2,9*0.D+0,5.528935335D-2,-2.336365955D-1,
     B  3.697071420D-1,-2.596415470D-1,6.828087013D-2/,
     *  C010,C011,C012,C310
     +  /1.94129239D-2,-1.69470576D-3,-4.311577033D+0,8.32875413D+0/,
C
C     THERMODYNAMIC SUB-REGION 4
C
     *  D
C    1  /2*0.D+0,-1.717616747D+0,1.301023613D+0,3.426663535D-4,
     1  /5*0.D+0,
     2  2*0.D+0,3.526389875D+0,-2.642777743D+0,-1.236521258D-3,
     3  2*0.D+0,-2.690899373D+0,1.996765362D+0,1.155018309D-3,2*0.D+0,
     4  9.070982605D-1,-6.661557013D-1,3*0.D+0,-1.138791156D-1,
     5  8.270860589D-2,0.D+0/
      DATA
C
C     PSEUDOCRITICAL DENSITY OF HEAVY WATER TO BE USED IN THE
C     CALCULATION OF THERMAL CONDUCTIVITY AND VISCOCITY
C
     *  PSEUDODC/0.358D+0/
      DATA
     *  JPC1
     1  /2,3,6,7,10,11,14,15,18,19,22,23,26,27,30,31,34,
     2  35,38,39,42,43,46,47,50,51,54,55,58,59,62,63/,
     *  JPC2
     1  /4,5,6,7,12,13,14,15,20,21,22,23,28,29,30,31,36,
     2  37,38,39,44,45,46,47,52,53,54,55,60,61,62,63/,
     *  JPC3
     1  /8,9,10,11,12,13,14,15,24,25,26,27,28,29,30,31,40,
     2  41,42,43,44,45,46,55,56,57,58,59,60,61,62,63/,
     *  JPC4
     1  /16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,
     2  48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63/
      END
```

Η μετατροπή της υπορουτίνας TOTAL από FORTRAN σε περιβάλλον προγραμματισμού MATLAB και SCILAB, χωρίς τη χρήση GOTO παρουσίασε σημαντική δυσκολία. Επιλέχθηκε να χρησιμοποιηθούν υποπρογράμματα function για κάθε εντολή GOTO που ζητούσε να εκτελεσθεί μία διαφορετική επιλογή.

Η αντίστοιχη function total σε περιβάλλον προγραμματισμού MATLAB δίνεται πιοκάτω (βλ. και στο Παράρτημα):

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
      A,MU,K,PRANDTL,SIGMA]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
      Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
%     IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
%     MUST BE CALLED TWICE.
%


%
%     ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
%     SPECIFICATION,JP
%
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 42 43 46 ...
      47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44 45 46 ...
      47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44 45 ...
      46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51 52 53 ...
      54 55 56 57 58 59 60 61 62 63];
[D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC]=deal(0.);
[GAMMA A MU K PRANDTL SIGMA]=deal(0.);
if mod(JP,2)<0
      T60;
end
if mod(JP,2)==0
      T70;
      T100;
end
if mod(JP,2)>0
      T60;
end
%
%     VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
%     TEMPERATURE AND INTERNAL ENERGY
%
function[]=T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70;
return
end
function[]=T70
```

```
for I=1:32
      if JP-JPC1(I)<0
      T100;
      return
      end
      if JP-JPC1(I)==0
      T90;
      return
      end
end
return
end
%
%     ENTHALPY AND ENTROPY
%
function[]=T90
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
T100;
return
end
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120;
      return
      end
end
return
end
%
%     SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT
VOLUME,
%     SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
%     MODULUS AND JOULE-THOMSON COEFFICIENT
%
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130;
return
end
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150;
      return
```

```
      end
end
return
end
%
%     VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
%
function[]=T150
[MU]=viscosity(TR,DD);
[K]=conductive(TR,DD);
if K~=0.D+0
      PRANDTL=1000.*MU*CP/K;
end
T160;
return
end
function[]=T160
for I=1:32
      if JP-JPC4(I)<0
      T190;
      return
      end
      if JP-JPC4(I)==0
      T180;
      return
      end
end
return
end
%
%     surface tension
%
function[]=T180
if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
end
T190;
return
end
function[]=T190
      if JP-32<0
      T200;
      return
      end
      if JP-32==0
      T210;
      return
      end
      if JP-32>0
      T210;
      return
      end
return
end
      function[]=T200
      return
      end
```

```
      function[]=T210
      return
      end
end
```

Η αντίστοιχη function total σε περιβάλλον προγραμματισμού SCILAB δίνεται πιοκάτω (βλ. και στο Παράρτημα):

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
A,MU,KK,PRANDTL,SIGMA]=total(JP,J,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
//    IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
//    MUST BE CALLED TWICE.
//
//    ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
//    SPECIFICATION,JP
//
exec('F:\scilab_hw\transient.sci');
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 42 43 46 ...
      47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44 45 46 ...
      47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44 45 ...
      46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51 52 53 ...
      54 55 56 57 58 59 60 61 62 63];
D=0.; B2=0.; B3=0.; B4=0.; DB2=0.; DB3=0.;
DB4=0.; U=0.; PSI=0.; H=0.; S=0.; CP=0.;
CV=0.; IBM=0.;JTC=0.; GAMMA=0.; A=0.;
MU=0.; K=0.; PRANDTL=0.; SIGMA=0.;
//
//    VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
//    TEMPERATURE AND INTERNAL ENERGY
//
function T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70();
return
endfunction
function[]=T70
for I=1:32
      if JP-JPC1(I)<0
      T100();
      return
      end
      if JP-JPC1(I)==0
      T90();
      return
      end
end
return
endfunction
//
//    ENTHALPY AND ENTROPY
//
function[]=T90
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
```

```
T100();
return
endfunction
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120();
      return
      end
end
return
endfunction
//
//    SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT VOLUME,
//    SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
//    MODULUS AND JOULE-THOMSON COEFFICIENT
//
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130();
return
endfunction
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150();
      return
      end
end
return
endfunction
//
//    VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
//
function[]=T150
[MU]=viscosity(TR,DD);
[KK]=conductive(TR,DD);
if KK~=0.D+0
exec('F:\scilab_hw\transient.sci');
      PRANDTL=1000.*MU*CP/KK;
end
T160();
return
endfunction
function[]=T160
for I=1:32
```

```
      if JP-JPC4(I)<0
      T190();
      return
      end
      if JP-JPC4(I)==0
      T180();
      return
      end
end
return
endfunction
//
//    surface tension
//
function[]=T180
      if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
      end
T190();
return
endfunction
function[]=T190
      if JP-32<0
      T200();
      return
      end
      if JP-32==0
      T210();
      return
      end
      if JP-32>0
      T210();
      return
      end
return
endfunction
      function[]=T200
      return
      endfunction
      function[]=T210
      return
      endfunction
      if pmodulo(JP,2)<0
      T60();
      end
      if pmodulo(JP,2)==0
      T70();
      T100();
      end
      if pmodulo(JP,2)>0
      T60();
      end
return
endfunction
```

Ο τρόπος μετατροπής της υπορουτίνας TOTAL από FORTRAN σε function total σε περιβάλλον προγραμματισμού MATLAB, παρουσιάσθηκε στο δικτυακό τόπο του MATLAB και μπορεί να βρεθεί κάτω από το σύνδεσμο

http://www.mathworks.com/matlabcentral/fileexchange/27591-a-multi-conditional-goto-like-implementation-in-matlab

ο οποίος μάλιστα γνωρίζει αρκετή μηνιαία επισκεψιμότητα (βλ. Pachitsas, 2010)

6.5 Μεταβλητές

Όπως έχει ήδη επισημανθεί στο 4^{ο} Κεφάλαιο, στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB είναι αποδεκτοί ως χαρακτήρες για τις παντός τύπου μεταβλητές και τα "κεφαλαία" και τα "πεζά". Μία μεταβλητή με όνομα σε "κεφαλαία" είναι διαφορετική από άλλη μεταβλητή με το ίδιο όνομα σε "πεζά". Αυτό γενικά δεν ισχύει στην FORTRAN, όπου δύο τέτοιες μεταβλητές ταυτίζονται. Όλες οι μεταβλητές στο περιβάλλον προγραμματισμού MATLAB είναι εξ'ορισμού πραγματικές "διπλής ακρίβειας". Στο περιβάλλον SCILAB όλες οι μεταβλητές είναι εξ'ορισμού μιγαδικές "διπλής ακρίβειας". Δεν μπορεί να γίνεται δήλωση του τύπου των μεταβλητών. σε αντίθεση με την FORTRAN, όπου υπάρχει αυτή η δυνατότητα. Στο MATLAB και στο SCILAB, το τι τύπου είναι οι μεταβλητές καθορίζεται τη στιγμή που αυτές παίρνουν τιμές. Αν επιζητείται η μετατροπή από τον εξ'ορισμού τύπο σε άλλο, π.χ. το πραγματικό μέρος ενός μιγαδικού, ή το ακέραιο μέρος ενός πραγματικού, διατίθενται για το σκοπό αυτό συγκεκριμένες function.

Για να εξηγηθεί το πώς χρησιμοποιούνται οι μεταβλητές στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, ορίζονται τα ακόλουθα επίπεδα ενός κώδικα:

Επίπεδο-(0): είναι το επίπεδο του κυρίως προγράμματος

Επίπεδο-(1): είναι τα επίπεδα των υποπρογραμμάτων που καλούνται από το κυρίως πρόγραμμα.

Επίπεδο-(2): είναι τα επίπεδα των υποπρογραμμάτων που καλούνται από τα υποπρογράμματα Επιπέδου-1

και γενικά

Επίπεδο-(n): είναι τα επίπεδα των υποπρογραμμάτων που καλούνται από τα υποπρογράμματα Επιπέδου-(n-1).

Οι μεταβλητές που χρησιμοποιούνται στο Επίπεδο-(n) παίρνουν την τιμή που έχουν στο Επίπεδο-(n-1). Για να γίνει αυτό αρκεί να έχουν το ίδιο όνομα και στα δύο Επίπεδα.

Οι μεταβλητές που παίρνουν τιμή στο Επίπεδο-(n) παραμένουν τοπικές σε αυτό το επίπεδο, ακόμα και αν έχουν το ίδιο όνομα με μεταβλητή του Επιπέδου-(n-1). Με άλλα λόγια, όταν ο έλεγχος επιστρέψει στο Επίπεδο-(n-1), η μεταβλητή παραμένει στην τιμή που είχε αρχικά στο Επίπεδο-(n-1).

Μεταβλητές που βρίσκονται στη λίστα παραμέτρων εισόδου υποπρογραμμάτων function, είναι στην ουσία τοπικές τόσο για το καλούμενο Επίπεδο-(n) όσο και για το καλόν Επίπεδο-(n-1). Οι τιμές τους είναι αυτές του Επιπέδου-(n-1).

Μεταβλητές που βρίσκονται στη λίστα παραμέτρων εξόδου υποπρογραμμάτων function, είναι στην ουσία τοπικές τόσο για το καλούμενο Επίπεδο-(n) όσο και για το καλόν Επίπεδο-(n-1). Οι τιμές τους είναι οι τιμές που πήραν (αν πήραν) στο Επίπεδο-(n).

Για να παραβιασθούν αυτοί οι γενικοί κανόνες πρέπει να χρησιμοποιηθούν συγκεκριμένες εντολές ως εξής:

(α) κατασκευάσθηκε ένα κείμενο εντολών (script), με το όνομα constants, στο αρχείο constants.m (ή constants.sci ανάλογα), το οποίο δίνει τιμές σε μεταβλητές που είναι σταθερές οι οποίες είναι γενικά χρήσιμες σε όλα τα επίπεδα 0,…, n, n+1,… των κωδίκων. Οι μεταβλητές αυτές ορίζονται ως καθολικές μεταβλητές των κωδίκων με την εντολή global. Το script αυτό καλείται σε όποια από τα επίπεδα είναι αναγκαίο.

(β) κατασκευάσθηκε ένα κείμενο εντολών (script), με το όνομα transient, στο αρχείο transient.m (ή transient.sci ανάλογα), το οποίο ορίζει ως καθολικές (global) τις μεταβλητές των κωδίκων που αλλάζουν τιμή σε διάφορα επίπεδα των κωδίκων και χρησιμοποιούνται σε άλλα επίπεδα των κωδίκων. Το script αυτό καλείται σε όποια από τα επίπεδα είναι αναγκαίο.

(γ) κατασκευάσθηκε ένα κείμενο εντολών (script), με το όνομα initial, στο αρχείο initial.m (ή initial.sci ανάλογα), το οποίο δίνει αρχικές τιμές 0 σε μεταβλητές που αλλάζουν τιμή σε διάφορα επίπεδα των κωδίκων. Πρόκειται σχεδόν για το σύνολο των

μεταβλητών που υπάρχουν στο script transient. Το script αυτό καλείται σε όποια από τα επίπεδα είναι αναγκαίο

Φυσικά, τα script (α), (β) είναι έτσι οργανωμένα, ώστε να καλύπτουν όλες τις μεταβλητές των κωδίκων, των οποίων οι τιμές πρέπει κάποια στιγμή να μεταφερθούν μεταξύ επιπέδων που δεν είναι διαδοχικά. Δεν υπάρχει επομένως δυνατότητα επιλογής της μεταβλητής ή έστω ενός υποσυνόλου των μεταβλητών που πρέπει να μεταφερθούν. Αυτή η δυνατότητα θα υπήρχε μόνο αν είχαν δημιουργηθεί περισσότερα κείμενα εντολών της μορφής constants και transient, τα οποία να περιέχουν υποομάδες των μεταβλητών αυτών.

Για την αποφυγή της μη επιθυμητής επικοινωνίας τιμών μεταβλητών από το Επίπεδο-(n) στο (n+1) επιλέχθηκε όλες οι μεταβλητές των κωδίκων να έχουν διαφορετικό όνομα.

Οι γενικές αρχές επικοινωνίας μεταξύ των επιπέδων κώδικα όπως ορίσθηκαν πιο πάνω, ισχύουν φυσικά και στη γλώσσα FORTRAN. Στους αντίστοιχους κώδικες σε FORTRAN το script constants είναι στην ουσία το υποπρόγραμμα BLOCKDATA που παρουσιάσθηκε στην προηγούμενη παράγραφο. Το ποιες από τις σταθερές αυτές είναι προς μεταφορά κατά υποομάδες, ορίζεται με εντολές COMMON. Σε όποιο επίπεδο των κωδίκων χρειάζεται να κληθεί μία υποομάδα σταθερών χρησιμοποιείται και η αντίστοιχη COMMON εντολή. Επίσης στους κώδικες σε FORTRAN οι μεταβλητές των κωδίκων που αλλάζουν τιμή σε διάφορα επίπεδα των κωδίκων και χρησιμοποιούνται σε άλλα επίπεδα των κωδίκων μεταφέρονται επίσης σε κατάλληλες υποομάδες με COMMON εντολές επικοινωνίας.

6.6 Τιμές στοιχείων πινάκων

Στη γλώσσα FORTRAN, ένας μονοδιάστατος πίνακας, θεωρείται πίνακας – στήλη και όχι πίνακας – γραμμή. Ομοίως ένας n-διάστατος, θεωρείται πίνακας n στηλών και όχι πίνακας n γραμμών. Όταν αποδίδονται τιμές στα στοιχεία ενός τέτοιου πίνακα, στην FORTRAN, τότε αυτό γίνεται κατά στήλες. Πρώτα δηλαδή, "γεμίζει" με δεδομένα η $1^{η}$ στήλη μετά η δεύτερη κοκ, μέχρι να εξαντληθεί η χωρητικότητα του πίνακα.

Αντίθετα στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB ένας μονοδιάστατος πίνακας, θεωρείται πίνακας – γραμμή και όχι πίνακας – στήλη. Ομοίως

ένας n-διάστατος, θεωρείται πίνακας n γραμμών και όχι πίνακας n στηλών. Όταν αποδίδονται τιμές στα στοιχεία ενός τέτοιου πίνακα, στο MATLAB και στο SCILAB, τότε αυτό γίνεται κατά γραμμές. Πρώτα δηλαδή, "γεμίζει" με δεδομένα η $1^{η}$ γραμμή μετά η δεύτερη κοκ, μέχρι να εξαντληθεί η χωρητικότητα του πίνακα. Ο τρόπος που αποδίδονται τιμές σε στοιχεία πινάκων στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, είναι πιο κοντά στις προσλαμβάνουσες παραστάσεις που έχει κάποιος για τη μορφή ενός πίνακα.

Στην περίπτωση που οι τιμές των στοιχείων ενός πίνακα πχ. του A, είναι διατεταγμένες κατά στήλες κατά τη λογική της FORTRAN και πρέπει να αναδιαταχθούν κατά γραμμές κατά τη λογική της MATLAB, πρέπει να χρησιμοποιηθεί η εντολή "transpose (A)". Στο περιβάλλον SCILAB, πρέπει απλά να γραφθεί η εντολή "A' = A". Με άλλα λόγια υπάρχουν εντολές στα περιβάλλοντα MATLAB και SCILAB για τον αυτόματο υπολογισμό του ανάστροφου πίνακα. Σε αυτό το σημείο πρέπει να τονισθεί ότι στο MATLAB και το SCILAB διατίθενται σημαντικός αριθμός εργαλείων για τον χειρισμό πινάκων που γενικά δεν υπάρχουν στη γλώσσα FORTRAN.

6.7 Εύρεση ρίζας εξίσωσης

Στη γλώσσα FORTRAN η εύρεση ρίζας εξίσωσης γίνεται συνήθως με κλήση κατάλληλης υπορουτίνας βιβλιοθήκης μαθηματικών και στατιστικών υποπρογραμμάτων. Η πλέον γνωστή και αξιόπιστη τέτοια βιβλιοθήκη είναι η λεγόμενη IMSL (International Mathematical and Statistical Library) της οποίας τα υποπρογράμματα διατίθενται επίσης για γλώσσες C, C++, Python κ.ά. Παρόλαυτά η βιβλιοθήκη IMSL δεν παρέχεται χωρίς κόστος κτήσης. Σε περίπτωση που δεν διατίθεται η βιβλιοθήκη IMSL, η εύρεση ρίζας εξίσωσης, μπορεί να γίνει με ανάπτυξη κατάλληλης υπορουτίνας σε γλώσσα FORTRAN από τον προγραμματιστή. Πράγματι, για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος, με κώδικες σε γλώσσα FORTRAN, όπου χρειάζεται η εύρεση ρίζας εξίσωσης, ο Πετρόπουλος (2003) χρησιμοποιεί την υπορουτίνα SOLVE, η οποία εφαρμόζει κατάλληλα μία αριθμητική μέθοδο NEWTON – RAPHSON.

Στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, κατασκευάσθηκε για τον ίδιο σκοπό αντίστοιχο υποπρόγραμμα function, το οποίο έχει το όνομα newton και λειτουργεί ακριβώς με τον ίδιο τρόπο. Το όνομα newton δόθηκε στο υποπρόγραμμα αυτό

διότι το όνομα solve είναι δεσμευμένο για άλλες εργασίες. Παρόλαυτά, στο MATLAB και το SCILAB διατίθενται έτοιμα σχετικά υποπρογράμματα που κάνουν ακριβώς την ίδια δουλειά και μάλιστα είναι δυνατόν να προσδιορίζουν ρίζες συστήματος εξισώσεων. Στο MATLAB η αντίστοιχη function που πρέπει να κληθεί, είναι η fzero, ενώ στο SCILAB η αντίστοιχη function που πρέπει να κληθεί, είναι η fsolve. Οι κώδικες υπολογισμού θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος που λειτουργούν με την fzero για MATLAB (ή την fsolve για SCILAB), δίνουν ακριβώς τα ίδια αποτελέσματα με εκείνους που χρησιμοποιούν την newton.

6.8 Διαδικασίες stop, return, end ή ισοδύναμες

Στη γλώσσα FORTRAN, με την εντολή STOP, ο έλεγχος επανέρχεται στον χρήστη, είτε αυτός εκφράζεται από την εξ'ορισμού είσοδο (δηλ. το τερματικό και το πληκτρολόγιο) είτε αυτός εκφράζεται από το περιβάλλον το οποίο κάλεσε το αντίστοιχο εκτελέσιμο πρόγραμμα.

Στα περιβάλλοντα MATLAB και SCILAB, δεν διατίθεται ακριβώς αντίστοιχη εντολή. Η κοντινότερη είναι η εντολή quit, η οποία όμως έχει το μεγάλο μειονέκτημα ότι όταν εκτελεσθεί οδηγεί το περιβάλλον MATLAB / SCILAB σε οριστικό "κλείσιμο", με αποτέλεσμα να εγκαταλείπεται το παράθυρο εντολών, στο οποίο πιθανόν και μέχρι τη στιγμή του quit να έχουν γραφθεί ενδιαφέροντα σχετικά μηνύματα από το πρόγραμμα προς τον χρήστη.

Όπως και στη γλώσσα FORTRAN έτσι και στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB η εντολή return, χρησιμοποιείται για την επιστροφή του ελέγχου από ένα υποπρόγραμμα, στο καλόν υποπρόγραμμα ή στο καλόν script. Επιπλέον, στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, η εντολή return αν χρησιμοποιηθεί ως εντολή μέσα σε script, επιστρέφει τον έλεγχο στο παράθυρο εντολών.

Στη γλώσσα FORTRAN, με την εντολή END, δηλώνεται το λογικό τέλος ενός κυρίως προγράμματος ή ενός υποπρογραμμάτος. Στο περιβάλλον MATLAB, (α) το λογικό τέλος ενός κυρίως προγράμματος (στην ουσία ενός κειμένου εντολών – script) δεν δηλώνεται και (β) το λογικό τέλος των υποπρογραμμάτων function δηλώνεται με end. Στο περιβάλλον SCILAB, (α) το λογικό τέλος ενός κυρίως προγράμματος (στην ουσία ενός

κειμένου εντολών – script) επίσης δεν δηλώνεται και (β) το λογικό τέλος των υποπρογραμμάτων function δηλώνεται με endfunction.

Στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, η εντολή end, χρησιμοποιείται για τερματισμό ενός if ελέγχου και των επαναληπτικών διαδικασιών for.

6.9 Δημιουργία εκτελέσιμου

Σε περιβάλλον FORTRAN ένας κώδικας εκτελείται μόνο αν δημιουργηθεί με κατάλληλο τρόπο αντίστοιχο εκτελέσιμο πρόγραμμα. Προκειμένου να γίνει αυτό, διατίθενται κατάλληλες εντολές με τις οποίες γίνονται τα ακόλουθα: (α) η συνολική μετάφραση (compilation) του πηγαίου κώδικα, (β) η σύνδεση (linking) μεταφρασμένων τμημάτων μεταξύ τους και τέλος (γ) η δημιουργία του αντίστοιχου εκτελέσιμου προγράμματος.

Στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, η δημιουργία εκτελέσιμου δεν είναι απαραίτητη. Οι εντολές που πρέπει να εκτελεσθούν, μπορούν να μεταφράζονται σειριακά μία-μία και να εκτελούνται αντίστοιχα. Αυτός είναι ο τρόπος που εκτελούνται προγράμματα που έχουν γραφθεί για γλώσσες, στις οποίες δεν υποστηρίζεται η διαδικασία συνολικής μετάφρασης (compilation) αλλά μόνο η διαδικασία σειριακής μετάφρασης και εκτέλεσης εντολών (interpreting). Αυτός ο τρόπος εκτέλεσης έχει το πρόβλημα της πολύ μειωμένης ταχύτητας ολοκλήρωσης των υπολογισμών, ιδιαίτερα για κώδικες όπως αυτοί της παρούσας Διπλωματικής Εργασίας που χρειάζονται πολλούς ελέγχους υποθέσεων και πολλές επαναλήψεις.

Για να αντιμετωπισθεί το πρόβλημα της αργής εκτέλεσης, στο περιβάλλον προγραμματισμού MATLAB, διατίθεται η δυνατότητα συνολικής μετάφρασης και παραγωγής εκτελέσιμου. Αν λοιπόν ένας κώδικας MATLAB αποτελείται από το script του κυρίως προγράμματος, από 1,…, n άλλα script και από 1,…, m άλλα υποπρογράμματα function, τότε η δημιουργία εκτελέσιμου γίνεται στο command window με χρήση της εντολής:

```
mcc –m main_program_script.m script1.m script2.m … scriptn.m
function1.m function2.m … functionm.m
```

όπου

*.m τα αρχεία που περιέχουν τις εντολές MATLAB, τα οποία στο συγκεκριμένο παράδειγμα πρέπει να βρίσκονται στο ίδιο directory.

Το δημιουργούμενο εκτελέσιμο, στην περίπτωση αυτή, θα πάρει το όνομα:

main_program_script.exe

και θα τοποθετηθεί στο directory που βρίσκονται και τα υπόλοιπα τμήματα του κώδικα. Για άλλες επιλογές σχετικά με την εντολή mcc, θα πρέπει κανείς να μελετήσει το σχετικό εγχειρίδιο του MATLAB. Το εκτελέσιμο που δημιουργείται, μπορεί να εκτελεσθεί κατά τον γνωστό τρόπο σε περιβάλλον MS Windows XP ή νεότερο ("double click" με το ποντίκι), αρκεί στον Η/Υ που θα γίνει αυτό, να υπάρχει εγκατεστημένη η κατάλληλη έκδοση MATLAB.

Για να αντιμετωπισθεί το πρόβλημα της αργής εκτέλεσης, στο περιβάλλον προγραμματισμού SCILAB, διατίθεται κάποια παρόμοια μεθοδολογία, η οποία δεν είναι τόσο απλή και κατανοητή όσο η εντολή mcc του MATLAB. Συγκεκριμένα, φαίνεται ότι κατ'αρχήν το κυρίως πρόγραμμα πρέπει να είναι σε C/C++ για να μπορέσει να γίνει η μετάφραση και η παραγωγή του εκτελέσιμου. Αυτό σημαίνει ότι θα πρέπει να υπάρχει κατ'αρχήν εγκατεστημένη γλώσσα C ή C++, προκειμένου να δημιουργηθεί το ζητούμενο εκτελέσιμο. Σε κάθε περίπτωση δεν διερευνήθηκε διεξοδικά στα πλαίσια της παρούσης Διπλωματικής Εργασίας, το πώς ακριβώς μπορεί αυτό να γίνεται. Υπολογίζεται ότι σε επόμενη έκδοση του SCILAB, η διαδικασία αυτή θα απλοποιηθεί σημαντικά.

6.10 Διαδικασίες debugging

Για γλώσσες FORTRAN εγκατεστημένες σε περιβάλλον LINUX, το debugging κωδίκων γίνεται με το γνωστό και φιλικό εργαλείο gdb. Τόσο οι γλώσσες FORTRAN όσο και το gdb για LINUX είναι ελεύθερα διαθέσιμα. Για γλώσσες FORTRAN, όπως αυτή της INTEL ή εκείνη της COMPAQ εγκατεστημένες σε περιβάλλον MS Windows XP ή νεότερο, διατίθενται ενσωματωμένοι debuggers. Παρόλαυτά αυτές οι γλώσσες FORTRAN δεν διατίθενται ελεύθερες και πρέπει να αγοράζονται. Για άλλες γλώσσες FORTRAN, που είναι ελεύθερα διαθέσιμες για περιβάλλον MS Windows XP ή νεότερο, υπάρχει συνήθως λογισμικό για debugging, αλλά δεν είναι σε όλες τις περιπτώσεις εύκολο να εγκατασταθεί και να χρησιμοποιηθεί.

Στο περιβάλλον MATLAB διατίθεται ενσωματωμένος debugger, πολύ κοντά στον τρόπο λειτουργίας του εργαλείου gdb. Στο περιβάλλον SCILAB, δεν διατίθεται ενσωματωμένος debugger. Αν το λογισμικό SCILAB είναι εγκατεστημένο σε LINUX, το εργαλείο gdb, μπορεί να χρησιμοποιείται και για το debugging κωδίκων σε SCILAB αρκεί οι κώδικες να καλούνται από κυρίως πρόγραμμα σε C και το σύνολο να μεταφράζεται προς εκτελέσιμο με τον διακόπτη –debug του μεταφραστή gcc. Υπολογίζεται ότι σε επόμενη έκδοση του SCILAB, η διαδικασία debugging θα απλοποιηθεί σημαντικά.

6.11 Σχόλια και συμπεράσματα

Σε αυτό το Κεφάλαιο, παρουσιάσθηκε ο τρόπος με τον οποίο αντικαταστάθηκαν ορισμένα από τα προγραμματιστικά εργαλεία που υπάρχουν στη γλώσσα FORTRAN από αντίστοιχα προγραμματιστικά εργαλεία που διατίθενται στα περιβάλλοντα προγραμματισμού MATLAB και SCILAB σε διακριτές παραγράφους για κάθε μία από τις σημαντικές διαφορές που αναφέρθηκαν στην εισαγωγή. Όσο αφορά στην ιδιομορφία του κυρίως προγράμματος, ένας κώδικας MATLAB ή SCILAB μπορεί να ξεκινήσει πρακτικά από οποιοδήποτε αρχείο κειμένου εντολών (script). Στο MATLAB και το SCILAB διατίθεται ένας και μόνο τύπος υποπρογράμματος. Με κατάλληλη χρήση του, ο τύπος αυτός μπορεί να αντικαταστήσει πολλούς τύπους υποπρογραμμάτων που συναντώνται σε άλλες γλώσσες. Αν αυτό δεν μπορεί να γίνει αποδοτικά είναι δυνατόν συγκεκριμένες εργασίες που σε άλλες γλώσσες επιλέγεται να γίνονται με υποπρογράμματα, στο MATLAB και το SCILAB να γίνονται με αρχεία κειμένων εντολών (script). Η εντολή GOTO είναι παντελώς ασύμβατη με την έννοια του δομημένου προγραμματισμού όπως υπηρετείται στο MATLAB και το SCILAB. Παρόλαυτά βρέθηκε ένας αποδοτικός τρόπος ώστε με εντολές του MATLAB ή του SCILAB, να λειτουργήσει το υποπρόγραμμα TOTAL, το οποίο στη γλώσσα FORTRAN είχε μόνο εντολές GOTO, για την κατευθυνόμενη μεταφορά του ελέγχου του κώδικα με βάση εντολές if. Σε ό,τι αφορά τον τρόπο που ορίζονται και μεταφέρονται από σημείο σε σημείο ενός κώδικα οι τιμές των καθολικών και των τοπικών μεταβλητών λήφθηκαν τα κατάλληλα μέτρα ώστε για τους κώδικες που αναπτύχθηκαν, να υπάρχει το ίδιο αποτέλεσμα που υπάρχει για τους αντίστοιχους κώδικες σε γλώσσα FORTRAN. Προσοχή χρειάζεται στον τρόπο που πληρώνονται οι θέσεις ενός πολυ-διάστατου πίνακα

με τιμές. Η πλήρωση γίνεται με διαφορετικό τρόπο από τη γλώσσα FORTRAN. Ο προσδιορισμός ρίζας εξίσωσης είναι πάρα πολύ ευκολότερος στο περιβάλλον MATLAB ή SCILAB, διότι διατίθενται ειδικά για το σκοπό αυτό υποπρογράμματα. Σε γλώσσα FORTRAN τα υποπρογράμματα αυτά, πρέπει είτε να κατασκευασθούν είτε να αγορασθούν χωριστά. Υπάρχουν λεπτές διαφορές στον τρόπο που λειτουργούν οι εντολές stop, return, end ή ισοδύναμες στο περιβάλλον MATLAB ή SCILAB σε σχέση με τη γλώσσα FORTRAN. Ο τρόπος που παράγεται εκτελέσιμος κώδικας είναι σχετικά εύκολος για το περιβάλλον MATLAB και κρίνεται χωρίς να έχει δοκιμασθεί, μάλλον όχι και τόσο φιλικός για το περιβάλλον SCILAB. Τέλος, οι διαδικασίες debugging είναι φιλικές για το περιβάλλον MATLAB και μάλλον δύσχρηστες για το περιβάλλον SCILAB.

ΕΠΙΛΟΓΟΣ

Στην παρούσα Διπλωματική Εργασία αναπτύχθηκαν με επιτυχία κώδικες σε περιβάλλοντα προγραμματισμού MATLAB και SCILAB για τον υπολογισμό των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος για διάφορες θερμοκρασίες και πιέσεις που καλύπτουν την περιοχή ισχύος των χρησιμοποιούμενων σχετικών εξισώσεων. Σε πίνακες που παρουσιάζονται στο τέλος του Παραρτήματος δίνονται χαρακτηριστικά αποσπάσματα υπολογισμού των θερμοφυσικών ιδιοτήτων του ελαφρού και του βαρέος ύδατος από τους κώδικες αυτούς. Διαπιστώνεται πολύ ικανοποιητική σύμπτωση. Συγκεκριμένα οι υπολογισμοί γίνονται στην περιοχή του υπέρθερμου υδρατμού, για τρεις πιέσεις 4.0, 4.2, και 4.4 bar και 25 θερμοκρασίες (από 250 έως 490 °C, βήμα 10 °C). Περιλαμβάνονται μόνο τα αποτελέσματα για τον ειδικό όγκο, την ειδική ενθαλπία και την ειδική εσωτερική ενέργεια. Τα αποτελέσματα συγκρίνονται με αντίστοιχους πίνακες για το ελαφρύ νερό όπως αυτοί δίνονται από τους τον Schmidt, 1987. και με αντίστοιχους πίνακες για το βαρύ νερό, όπως αυτοί δίνονται από τον Hill, 1981. Οπως παρατηρείται, τα αποτελέσματα των υπολογισμών με εκείνα των πινάκων συμφωνούν ικανοποιητικά. Τα κυρίως προγράμματα με βάση τα οποία λήφθηκαν τα αποτελέσματα αυτά δίνονται επίσης στην ίδια θέση του Παραρτήματος.

Τα περιβάλλοντα προγραμματισμού MATLAB και SCILAB, είναι σήμερα (2010) πολύ δημοφιλή και εξαιρετικά φιλικά προς τον χρήστη, διότι διαθέτουν μαθηματικά και γραφικά εργαλεία σε πολύ μεγαλύτερη έκταση από οποιοδήποτε άλλο σχεδόν προγραμματιστικό περιβάλλον. Επιπλέον ειδικά το SCILAB παρέχεται χωρίς κόστος, πράγμα που αντισταθμίζει την περιορισμένη του επέκταση, ορισμένες λιγότερες ευκολίες καθώς επίσης και την περιορισμένη τεκμηρίωση σε σχέση με το MATLAB. Επομένως η δοκιμή δημιουργίας κωδίκων θερμοφυσικών ιδιοτήτων σε αυτά τα περιβάλλοντα έπρεπε να γίνει για να διαπιστωθεί η χρηστικότητα τους σε σχέση και με αυτό το ζήτημα υπολογισμών. Σημειώνεται ότι, ειδικά σε ό,τι αφορά το MATLAB, τα εκτελέσιμα που μπορούν να δημιουργηθούν από τους κώδικες είναι μεταφέρσιμα σε άλλους υπολογιστές που διαθέτουν ίδια ή κοντινή έκδοση MATLAB και μπορούν να εκτελεσθούν και εκεί χωρίς να χρειάζεται να διατεθεί ο αντίστοιχος πηγαίος κώδικας. Είναι δυνατόν στο μέλλον να πραγματοποιηθεί η εκμετάλλευση ορισμένων επιπλέον δυνατοτήτων που

δίνονται από το MATLAB και το SCILAB, σε σύγκριση με τις δυνατότητες κωδίκων που έχουν γραφθεί σε γλώσσα FORTRAN. Αυτές οι δυνατότητες είναι κυρίως:

- η μετατροπή ενός κώδικα σε γλώσσα java, η οποία μπορεί να εκτελείται αποδοτικά ανεξάρτητα από Λειτουργικό Σύστημα και υλικό.
- η παραγωγή dll, το οποίο μπορεί να καλείται από το πρόγραμμα του χρήστη σε περιβάλλον MS Windows XP
- η κλήση υποπρογραμμάτων σε C ή FORTRAN από περιβάλλον MATLAB και SCILAB
- η κλήση κώδικα σε MATLAB ή SCILAB από γλώσσα C ή FORTRAN

και τέλος

- η χρήση των εξελιγμένων γραφικών δυνατοτήτων

Οι δυνατότητες αυτές μπορούν κατ'αρχήν να διερευνηθούν σε επόμενη Διπλωματική Εργασία, με αντικείμενο φυσικά κώδικες που αφορούν στην Πυρηνική Τεχνολογία, με τους ακόλουθους περιορισμούς:

(1) Για το περιβάλλον MATLAB η διερεύνηση μπορεί να γίνει ανεξάρτητα από Λειτουργικό Σύστημα. Θα πρέπει όμως, για το σκοπό αυτό να υπάρχουν διαθέσιμες γλώσσα FORTRAN και γλώσσα C της INTEL, οι οποίες είναι συμβατές με το περιβάλλον MATLAB. Αυτές οι εκδόσεις των γλωσσών αυτών δεν είναι ελεύθερα διαθέσιμες και έχουν κόστος αγοράς (~ 2000 EUR και οι δύο 2010). Ειδικά για την παραγωγή java κώδικα θα πρέπει να υπάρχει εγκατεστημένο το κατάλληλο java περιβάλλον όπως διατίθεται δωρεάν.

(2) Για το περιβάλλον SCILAB η διερεύνηση μπορεί να γίνει καλύτερα σε Λειτουργικό Σύστημα LINUX, για το οποίο φαίνεται ότι αρχικά δημιουργήθηκε το SCILAB. Θα πρέπει, για το σκοπό αυτό να υπάρχουν διαθέσιμες γλώσσα FORTRAN και γλώσσα C. Αυτές οι εκδόσεις των γλωσσών αυτών είναι ελεύθερα διαθέσιμες για Λειτουργικό Σύστημα LINUX.

Κλείνοντας, είναι σημαντικό να αναφερθεί ότι οι μεταφορές κωδίκων από γλώσσες προγραμματισμού που δεν είναι καθαρά $4^{ης}$ γεννεάς, όπως είναι η FORTRAN, σε

γλώσσες προγραμματισμού $4^{ης}$ γεννεάς όπως είναι το περιβάλλον MATLAB και το SCILAB, αντιμετωπίζει προβλήματα. Δεδομένου ότι οι κώδικες που παρουσιάσθηκαν εδώ προέρχονται από μετατροπή άλλων πρωτότυπων σε FORTRAN, ένα από τα σημαντικότερα που αντιμετωπίσθηκε κατά την μετατροπή ήταν η έλλειψη εντολών GOTO στο MATLAB και το SCILAB. Ο τρόπος με τον οποίο ξεπεράσθηκε αυτό το εμπόδιο σε συγκεκριμένο τμήμα των κωδίκων, δημοσιεύθηκε στο δικτυακό τόπο της Mathworks, δηλαδή της εταιρείας κατασκευής του MATLAB, και καταγράφει ήδη σημαντικό αριθμό επισκέψεων ανά μήνα στον ακόλουθο σύνδεσμο:

http://www.mathworks.com/matlabcentral/fileexchange/27591-a-multi-conditional-goto-like-implementation-in-matlab

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Garland W.J., Hoskins J.D., "Approximate Functions for the Fast calculation of Light-Water Properties at Saturation", International Journal of Multiphase Flow, 14(3):333-348, 1988
2. Garland W.J., Hand B.J., "Simple Functions for the Fast Approximation of Light-Water Thermodynamic Properties", Nuclear Engineering and Design, 113:21-34, 1989
3. Garland W.J., Wilson R.J., Bartak J., Cizek J., Stastny M., Zentrich I., "Extensions to the Approximation Function for the Fast calculation of Saturated Water Properties", Nuclear Engineering and Design, 136:381-388, 1992
4. Haar L., Gallagher J.S., Kell G.S, "NBS/NRC Steam Tables – Thermodynamic and Transport Properties and Computer Programs for vapor and Liquid States of Water in SI Units", Hemisphere Publishing Corporation, Washington D.C., 1984.
5. Hendricks R.C., Peller R.C., Baron A.K., "WASP – A Flexible FORTRAN IV Computer Code for Calculating Water and Steam Properties", Report No. NASA Technical Note TN D-7391, NASA, Washington, D.C. 20546, USA 1973.
6. Hill P.G., MacMillan R.D.C., Lee V., "Tables of Thermodynamic Properties of Heavy Water in S.I. Units", Department of Mechanical Engineering, The University of British Columbia, Atomic Energy of Canada Limited, AECL-7531, 1981
7. Hill P.G., MacMillan R.D.C., Lee V., "A Fundamental Equation of State for Heavy Water", Journal of Physical Chemistry Reference Data, 11(1):1, 1982; Errata: 12(4):1065, 1983
8. Kelvin Hales Associates Ltd, "Water and Steam Thermophysical Properties for MATLAB – Extract from the User's manual", http://www.khace.com/products/watsteam/files/wsxtract.pdf, Surrey, 2001.
9. Keyes F.G., Keenan J.H., Hill P.G., Moore J.G., "A Fundamental Equation for Liquid and Vapor Water", Proceedings of the 7th International Conference on the Properties of Steam, Tokyo 1968.
10. Κωστάκος Γ., "Κώδικας Υπολογισμού Θερμοδυναμικών Ιδιοτήτων και Ιδιοτήτων Μεταφοράς του Βαρέος Υδατος", Διπλωματική Εργασία, Τομέας Πυρηνικής Τεχνολογίας ΕΜΠ, Αθήνα 1986

11. Matsunaga N., Nagashima A., "Transport Properties of Liquid and Gaseous Deuterium Oxide, over a wide Range of Temperature and Pressure", Journal of Physical Chemistry Reference Data, 12(4):933, 1983
12. Pachitsas S.T., Petropoulos N.P., "A multi conditional goto-like Implementation in MATLAB", http://www.mathworks.com/matlabcentral/fileexchange/27591-a-multi-conditional-goto-like-implementation-in-matlab, 2010
13. Πετρόπουλος Ν.Π., "Βιβλιογραφική Αναζήτηση και Κριτική Ανασκόπηση Σχέσεων για τις Θερμοφυσικές Ιδιότητες του Βαρέος Ύδατος – Κώδικας Υπολογισμών HEAVY_WASP", Διπλωματική Εργασία, Τομέας Πυρηνικής Τεχνολογίας ΕΜΠ, Αθήνα 1991
14. Πετρόπουλος Ν.Π., "Θερμοφυσικές Ιδιότητες Ψυκτικών Μέσων Πυρηνικών Αντιδραστήρων – Συγκριτική Μελέτη και Κώδικες Υπολογισμού", Διδακτορική Διατριβή, Τομέας Πυρηνικής Τεχνολογίας ΕΜΠ, Αθήνα 2003.
15. Schmidt E., Grigull U., "Πίνακες Ιδιοτήτων Νερού-Υδρατμού, Διεθνές Σύστημα Μονάδων", Μετάφραση – Επιμέλεια: Ρακόπουλος Κ.Δ., Εκδοση: Φούντας Γ., Αθήνα, 1987.
16. Sengers J.V., Watson J.T.R., "Improved International Formulations for the Viscosity and Thermal Conductivity of Water Substance", Journal of Physical Chemistry Reference Data, 15(4):1291, 1986
17. Straub J., Rosner N., Grigull U., "Oberflaechenspannung von Leichtem und Schwerem Wasser", Waerme- und Stoffuebertragung, 13:241, 1980
18. Ulrych G., "Schweres Wasser-Dampftafel. Die 1967-IFC-Formulation fuer Industriegebrauch als Grundlage einer Dampftafel fuer schweres Wasser und Zusammenstellung der Transportgroessen", Technischer Bericht Nr. R 12-330/80, Fortschritt-Berichte der VDI Zeitschriften, Reihe 6 (Energietechnik-Waermetechnik), Nr. 90, Erlangen; VDI-Verlag GmbH Duesseldorf, Bundesrepublik Deutschland 1981
19. Wagner W., Kruse A., "Properties of Water and Steam, the Industrial Standard IAPWS-IF97 for the Thermodynamic Properties and Supplementary Equations for Other Properties", Springer-Verlag, Berlin, 1998

ΠΑΡΑΡΤΗΜΑΤΑ

ΚΕΙΜΕΝΑ ΚΩΔΙΚΩΝ

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Παράρτημα Ι: Κώδικας ελαφρού ύδατος Π-1

Ι.1 Περιβάλλον προγραμματισμού MATLAB Π-1

Ι.2 Περιβάλλον προγραμματισμού SCILAB.....Π-40

Παράρτημα ΙΙ: Κώδικας βαρέος ύδατος Π-80

ΙΙ.1 Περιβάλλον προγραμματισμού MATLAB.....Π-80

ΙΙ.2 Περιβάλλον προγραμματισμού SCILAB Π-119

Παράρτημα ΙΙΙ: Ενδεικτικά αποτελέσματα κωδίκων Π-159

ΙΙΙ.1 Κώδικας ελαφρού ύδατος - περιβάλλον MATLAB & SCILAB Π-159

ΙΙΙ.2 Απόσπασμα ιδιοτήτων από πίνακες ελαφρού ύδατος Π-161

ΙΙΙ.3 Κώδικας βαρέος ύδατος - περιβάλλον MATLAB & SCILAB Π162

ΙΙΙ.4 Απόσπασμα ιδιοτήτων από πίνακες βαρέος ύδατος.....Π-164

ΠΑΡΑΡΤΗΜΑ Ι: Κώδικας ελαφρού ύδατος

Ι.1 Περιβάλλον προγραμματισμού MATLAB

Τα υποπρογράμματα παρατίθενται κατά την αλφαβητική σειρά της ονομασίας τους. Συμπεριλαμβάνεται το κυρίως πρόγραμμα example.m.

Ι.1.1 Function checkpt (αρχείο checkpt.m)

```
function[TS,JR,IS]=checkpt
IER=0;
TS=0.;
JR=0;
IS=0;
constants;
transient;
EPS=1.D-5; NDEC=5; ITMAX=100;
%
%     CHECK PRESSURE FOR OUT OF RANGE
%
if (PP<PMIN | PP>PMAX) & PP~=0.
      disp(' ')
      disp('Desired Pressure out of range')
      disp(' ')
      return
end
%
%     CHECK TEMPERATURE FOR OUT OF RANGE
%
if (TT<TMIN | TT>TMAX) & (TT~=0.)
      disp(' ')
      disp('Desired Temperature out of range')
      disp(' ')
      return
end
%
%     IF PRESSURE AND TEMPERATURE ARE EQUAL TO 0 CHECK FOR
%     CORRECT CALL
%     OF SATURATION PROPERTIES
%
if PP==0. & TT==0.
      disp(' ')
      disp('Both pressure and temperature cannot be 0.!')
      disp(' ')
      return
end
%
%     IF PRESSURE OR TEMPERATURE IS EQUAL TO 0 CHECK FOR OUT
%     OF
%     SATURATION RANGE
%
if (PP>PC & TT==0.) | (TT>TC & PP==0.)
      disp(' ')
      disp('Such a saturation state does not exist!')
      disp(' ')
      return
end
if TT==0.;
      TT=TT+273.15D+0;
```

```
end
if PP==0.;
      [FPS]=fps(TT);
      PP=FPS*10;
      TS=TT;
      IS=1;
else
      if PP<=PC & TT<=TC
      [FTS]=fts();
      TS=FTS;
      [FPS]=fps(TS);
      if abs(FPS)>EPS
      f='fps';
      df='dfps';
      [TS,IER]=newton(f,df,EPS,NDEC,TS,ITMAX);
      IS=1;
      end
      if TT==273.15D+0
      TT=TS;
      end
      end
end
if abs(TT-TS)<=1.D+0
      IS=2;
end
%
%     REDUCED PRESSURE AND TEMPERATURE
%
PR=PP/PC;
TR=TT/TC;
%
%     DETERMINE THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
if TR>=TRT & TR<=TR1
      if abs(TT-TS)<=1.D-5
            JR=6;
            return
      else
            [PRS]=prs();
      if PR>=0. & PR<PRS
            R=2;
            return
      end
      if PR>PRS & PR<=PR2
            JR=1;
            return
      end
      end
end
if TR>=TR & ((TR-1)<1.D-5)
      if abs(TT-TS)<=1.D-5
            JR=5;
            return
            end
else
      [PRS]=prs();
      [PRL]=prl();
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
```

```
      if PR>PRL & PR<PRS
            JR=3;
            return
            end
      [PRS]=prs();
      if PR>PRS & PR<PR2
            JR=4;
            return
      end
end
      [PRL]=prl();
if (abs(TR-1)>=1.D-5) & TR<TR2
      if PR>=0. & PR<=PRL
            JR=2;
            return
            end
      if PR>PRL & PR<=PR2
            JR=3;
            return
            end
end
if TR>=TR2 & TR<=TR3
      JR=2;
      return
end
if JR==0
      disp(' ')
      disp('Internal Error: JR indicator value is zero')
      disp(' ')
      return
end
return
end
```

I.1.2 Function conductive (αρχείο conductive.m)

```
function [K]=conductive(TT,DD)
%
%     COMPUTE THERMAL CONDUCTIVITY OF LIGHT WATER IN W/(mK) GIVEN
%     TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%     FROM AN EQUATION SUITABLE FOR INDUSTRIAL USE
%
constants;
      A=[0.0102811D+0 0.0299621D+0 0.0156146D+0 -0.00422464D+0];
      B=[-0.397070D+0 0.400302D+0 1.060000D+0];
      B1=-0.171587D+0;
      B2=2.392190D+0;

      C1= 0.642857D+0;
      C2=-4.11717D+0;
      C3=-6.17937D+0;
      C4=0.00308976D+0;
      C5=0.0822994D+0;
      C6=10.0932D+0;

      D1=0.0701309D+0;
      D2=0.0118520D+0;
      D3=0.00169937D+0;
      D4=-1.0200D+0;
      ELC=1.D+0;
      PSEUDODR=DD/PSEUDODC;
      PSEUDOTR=TT/PSEUDOTC;
      ELR0=0.D+0;
for   I=1:4
      ELR0=ELR0+A(I)*(PSEUDOTR^(I-1));
end
      ELR0=sqrt(PSEUDOTR)*ELR0;
      ELR1=B(1)+B(2)*PSEUDODR+B(3)*exp(B1*((PSEUDODR+B2)^2));
      DELTATR=abs(PSEUDOTR-1.)+C4;
      Q=2.+C5/(DELTATR^0.6);
      R=Q+1.;
      S=1./DELTATR;
if PSEUDOTR<1.D+0
      S=C6/(DELTATR^0.6);
end
ELR2=(D1/(PSEUDOTR^10)+D2)*(PSEUDODR^1.8)*exp(C1*(1-(PSEUDODR^2.8)))...
+D3*S*(PSEUDODR^Q)*exp((Q/R)*(1-...
(PSEUDODR^R)))+D4*exp(C2*(PSEUDOTR^1.5)+C3/(PSEUDODR^5));
      EL=ELC*(ELR0+ELR1+ELR2);
      K=EL;
      return
end
```

I.1.3 Script constants.m

```
'constants';
%
%     SCRIPT CONTAINING ALL CONSTANTS OF LIGHT_WASP
%     BEING USED AT LEAST TWICE.ALL OTHER ONCE USED CONSTANTS ARE
%     PART OF CORRESPONDING ASSIGNMENTS IN INDIVIDUAL SUBPROGRAMS
%
global PMIN PMAX TMIN TMAX PC TC VC TRT TR1 TR2 TR3 PR1 PR2 R K C
global C010
global C011 C012 C310 D PSEUDODC PSEUDOTC PSEUDOPC JPC1 JPC2 JPC3
global JPC4
global ZERO
%
%     COMMONS LABELLED WITH Z REPRESENT ALL CONSTANTS OF LIGHT_WASP
%     BEING USED AT LEAST TWICE
%
%
%     PRESSURE AND TEMPERATURE RANGE OF EQUATIONS FOR THERMODYNAMIC
%     AND TRANPORT PROPERTIES OF LIGHT WATER
%
      PMIN=0.006D+0; PMAX=1000.D+0; TMIN=273.16D+0; TMAX=1273.15D+0;
%
%     CRITICAL PRESSURE,TEMPERATURE AND SPECIFIC VOLUME OF LIGHT
%     WATER
%
      PC=221.2D+0; TC=647.3D+0; VC=3.17D+0;
%
%     REDUCED TEMPERATURE RANGES OF THERMODYNAMIC SUB-REGIONS
%
      TRT=4.21999073D-1; TR1=9.626911787D-1; TR2=1.333462073D+0;
      TR3=1.657886606D+0;
%
%     MAXIMUM REDUCED PRESSURE
%
      PR1=7.475191707D-1; PR2=4.520795660D+0;
%
%     LIGHT WATER CONSTANT
%
      R=0.46151D+0;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS OF EQUATION OF STATE
%     OF LIGHT WATER BEING USED IN THIS PACKAGE
%
%
%     SATURATION LINE
%
      K=[-7.691234564D+0 -2.608023696D+1 ...
      -1.681706546D+2 6.423285504D+1 ...
      -1.189646225D+2 4.167117320D+0 2.097506760D+1 1.D+9 6.D+0];
%
%     THERMODYNAMIC SUB-REGIONS 3 AND 4
%
C=[-1.72260420D-2 -7.77175039D+0 4.20460752D+0 -2.76807038D+0 ...
      2.10419707D+0 -1.14649588D+0 2.23138085D-1 1.16250363D-1 ...
      -8.20900544D-2 0.D+0,7.08636085D-1 1.23679455D+1 ...
      -1.20389004D+1 5.40437422D+0 -9.93865043D-1 ...
      6.27523182D-2 -7.74743016D+0 0.D+0 0.D+0 0.D+0 ...
      -4.29885092D+0 4.31430538D+1 -1.41619313D+1 ...
      4.04172459D+0 1.55546326D+0 -1.66568935D+0 ...
```

```
      3.24881158D-1 2.93655325D+1 0.D+0 0.D+0 ...
      7.94841842D-6 8.0885947D+1 -8.36153380D+1 ...
      3.58636517D+1 7.51895954D+0 -1.26160640D+1 1.09717462D+0 ...
      2.12145492D+0 -5.46529566D-1 2.75971776D-6 -5.09073985D-4 ...
      0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      2.10636332D+2 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      0.D+0 0.D+0 5.528935335D-2 -2.336365955D-1 ...
      3.697071420D-1 -2.596415470D-1 6.828087013D-2];

      C010=1.94129239D-2;
      C011=-1.69470576D-3;
      C012=-4.311577033D+0;
      C310=8.32875413D+0;

%
%     THERMODYNAMIC SUB-REGION 4
%
D=[0.D+0 0.D+0 -1.717616747D+0 1.301023613D+0 3.426663535D-4;...
      0.D+0 0.D+0 3.526389875D+0 -2.642777743D+0 -1.236521258D-3;...
      0.D+0 0.D+0 -2.690899373D+0 1.996765362D+0 1.155018309D-3;...
      0.D+0 0.D+0 9.070982605D-1 -6.661557013D-1 0.D+0;...
      0.D+0 0.D+0 -1.138791156D-1 8.270860589D-2 0.D+0];
      D=transpose(D);

%     PSEUDOCRITICAL DENSITY AND TEMPERATURE OF LIGHT WATER TO BE
%     USED IN THE CALCULATION OF THERMAL CONDUCTIVITY AND VISCOCITY
%
      PSEUDODC=0.317363D+0; PSEUDOTC=647.27D+0; PSEUDOPC=221.15D+0;
```

I.1.4 Function densf (αρχείο densf.m)

```
function[DF,IER]=densf
%
%     COMPUTE DENSITY OF LIQUID HEAVY WATER IN g/cm3 GIVEN PRESSURE
%     IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
%     IS EQUAL TO 0 SATURATED LIQUID DENSITY IS RETURNED
%
IER=0;
constants;
transient;
EPS=1.D-5; NDEC=5; ITMAX=100;
DF=0; DL=0;
[DL,IER]=svlwl;
f='fp';
df='dfpd';
[DF,IER]=newton(f,df,EPS,NDEC,DL,ITMAX);
ZZ=DF;
qmust(ZZ);
return
end
```

I.1.5 Function densg (αρχείο densg.m)

```
function[DG,IER]=densg
%
%     COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 GIVEN PRESSURE
%     IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
%     IS EQUAL TO 0 SATURATED VAPOR DENSITY IS RETURNED
%
IER=0;
constants;
transient;
EPS=1.D-5; NDEC=5;ITMAX=100;
DG=0; DV=0;
[DV, IER]=svlwv;
FP='fp';
DFPD='dfpd';
[DG, IER]=newton(FP,DFPD,EPS,NDEC,DV,ITMAX);
DD=DG;
ZZ=DG;
qmust(ZZ);
return
end
```

I.1.6 Function dfpd (αρχείο dfpd.m)

```
function[DFPD]=dfpd(ZZ)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF LIGHT WATER PRESSURE BY
%     DENSITY
%
constants;
transient;
DFPD=R*TT*(1+2*ZZ*Q+4*ZZ*ZZ*QDT+ZZ*ZZ*ZZ*Q2D2T);
return
end
```

I.1.7 Function dfpr3 (αρχείο dfpr3.m)

```
function[DFPR3]=dfpr3(X3)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIGHT
%     WATER VAPOR BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
%     SUB-REGIONS 3 AND 5
%
constants;
transient;
DXSC0N=0.;
DXSC1N=0.;
DXSC2N=0.;
DXSC3N=0.;
DXSC4N=0.;
DXSC6N=0.;
for N=0:9
      if N>=2
            DXSC0N=DXSC0N+(N^2-N)*C(N)/(X3^(N+1));
      end
      if N>=2 & N<=6
            DXSC1N=DXSC1N+(N^2-N)*C(10+N)/(X3^(N+1));
      end
      if N>=2 & N<=7
            DXSC2N=DXSC2N+(N^2-N)*C(20+N)/(X3^(N+1));
      end
      if N>=2
            DXSC3N=DXSC3N+(N^2-N)*C(30+N)/(X3^(N+1));
      end
      if N<=4
            DXSC6N=DXSC6N+C(60+N)/(TR^(N+2));
      end
end
DXSC0N=DXSC0N+(100-10)*C010/(X3^11);
DXSC0N=DXSC0N+(121-11)*C011/(X3^12);
DXSC0N=DXSC0N-C012/(X3^2);
DXSC1N=DXSC1N-C(17)/(X3^2);
DXSC2N=DXSC2N-C(28)/(X3^2);
DXSC3N=DXSC3N-C310/(X3^2);
DXSC4N=-30*C(41)*(TR-1)/((X3^7)*(TR^23));
DXSC6N=30*(X3^4)*DXSC6N;
DFPR3=-DXSC0N-DXSC1N*(TR-1)-DXSC2N*((TR-1)^2)...
      -DXSC3N*((TR-1)^3)+DXSC4N-DXSC6N;
return
end
```

I.1.8 Function dfpr4 (αρχείο dfpr4.m)

```
function[DFPR4]=dfpr4(X4)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIQUID
%     LIGHT WATER BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
%     SUB-REGIONS 4 AND 5
%
constants;
transient;
Y=(1-TR)/(1-TR1);
DXSDN1=0.;
DXSDN2=0.;
DXSDMN=0.;
for M=3:4
      for N=1:5
            DXSDN1=DXSDN1-N*(N-1)*D(M,N)*(Y^M)/(X4^(N+1));
      end
      DXSDMN=DXSDMN+DXSDN1;
end
for N=1:3
      DXSDN2=DXSDN2+(N-1)*(N-2)*D(5,N)*(X4^(N-3));
end
DXSDN2=(Y^32)*DXSDN2;
[DFPR3]=dfpr3(X4);
DFPR4=DFPR3+DXSDMN-DXSDN2;
return
end
```

I.1.9 Function dfps (αρχείο dfps.m)

```
function[DFPS]=dfps(TS)
%
%     COMPUTE 1st DERIVATIVE OF LIGHT WATER SATURATION PRESSURE BY
%     TEMPERATURE
constants;
transient;
%
%     REDUCE SATURATION TEMPERATURE
%
      TR=TS/TC;
%
%     CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
%
      PPC=PC/10.;
      SUMKN=0.;
      DSUMKN=0.;
for N=1:5
            SUMKN=SUMKN+K(N)*((1-TR)^N);
            DSUMKN=DSUMKN+N*K(N)*((1-TR)^(N-1));
end
      PRK1=(1/TR)*SUMKN;
      PRK2=1+K(6)*(1-TR)+K(7)*((1-TR)^2);
      PRK3=K(8)*((1-TR)^2)+K(9);
      PRK4=PRK1/PRK2-(1-TR)/PRK3;
      FPS=exp(PRK4)*PPC;
      DPRK1=(-1/TR)*PRK1/PRK2;
      DPRK2=DPRK1+(-DSUMKN*PRK2+SUMKN*(K(6)+2*K(7)*...
      (1-TR)))/TR/(PRK2^2);
      DPS=DPRK2+(-PRK3+(1-TR)*(2*K(8)*(1-TR)))/(PRK3^2);
      DFPS=FPS*DPS/TC;
      return
end
```

I.1.10 Function energy (αρχείο energy.m)

```
function[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD)
%
%     COMPUTE SPECIFIC INTERNAL ENERGY OF LIGHT WATER IN kJ/kg GIVEN
%     TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
U=PSI0-TT*PSI0T+R*1000*DD*QTD;
PSI=PSI0+R*TT*(log(DD)+DD*Q);
return
end
```

I.1.11 Function enthalpy (αρχείο enthalpy.m)

```
function[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD)
%
%     COMPUTE ENTHALPY OF LIGHT WATER IN kJ/kg GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
H=R*TT*(1+DD*Q+DD*(1000/TT)*QTD+DD*DD*QDT)+PSI0-TT*PSI0T;
return
end
```

I.1.12 Function entropy (αρχείο entropy.m)

```
function[S]=entropy(TT,DD,PSI0T,Q,QTD)
%
%     COMPUTE ENTROPY OF LIGHT WATER IN kJ/(kgK) GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
S=-R*(log(DD)+DD*Q-DD*(1000/TT)*QTD)-PSI0T;
return
end
```

I.1.13 Κυρίως πρόγραμμα (script) example.m

```
'program example';
%
%MASTER TEST PROGRAM FOR HEAVY WATER PROPERTY PACKAGE
%
constants;
transient;
disp('National Technical University of Athens')
disp('Department of Mechanical Engineering')
disp('Nuclear Engineering Section')
disp(' ')
disp('LIGHT W.A.S.P. (R)')
disp('LIGHT WATER AND STEAM PROPERTIES PACKAGE')
disp('Thermodynamic and Transport Properties')
disp('of the light Water Substance;an')
disp('interactive approach')
disp('MATLAB Version 1-SEPTEMBER 2010')
disp('Author: STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>')
disp('This version of code accepts')
disp('as input only pressure and tempreture')
disp('(range: 0.006-1000bar,0.01-1000 C)')
disp('To obtain a saturation state, equal either')
disp('pressure or temperature to zero(for the')
disp('desired temperature or pressure,respectively')
disp(' ')
SUPER=false;
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
%
[PP PR TT TR DD JJ]=deal(0.);
disp(' ')
disp('Please enter desired pressure in bars:')
P=input('P:');
disp('Please enter desired temperature in C:')
T=input('T:');
if T~=0
      T=T+273.15;
end
JS=1;
JP=63;
[P,T,TS,SVF,SVG,JR,IS]=light_wasp(JS,JP,P,T);
if JR==0
      return
end
fprintf('Pressure in bars:%8.3f\n',P);
fprintf('Temperature in C:%8.3f\n',T-273.15);
disp(' ')
if P>PC & T>TC
      SUPER=true;
      disp('SUPERCRITICAL STATE ')
end
if P<=PC & SVF~=0 & IS~=2
      disp('SUBCOOLED LIQUID REGION ')
end
if P<=PC & SVG~=0 & IS~=2
      disp('SUPERHEATED STEAM REGION ')
end
if IS==0
      disp('There is no saturation state for this pressure and temperature')
```

```
      disp(' ')
      disp(' ')
end
if IS==1
      disp('Exact saturation temperature for this pressure')
      fprintf('%7.3f\n',TS-273.15);
      disp(' ')
end
if IS==2
      disp('This is a saturation state or almost a saturation state')
      disp(' ')
      disp(' ')
      disp(' ')
end
if SUPER == true
      disp('DENSE PHASE ')
if SVF>0
      disp(' ')
      fprintf('Specific Volume in m3/kg:%8.7f\n',SVF);
      fprintf('Internal Energy in KJ/Kg:%6.1f\n',UF);
      fprintf('Specific Enthalpy in KJ/Kg:%6.1f\n',HF);
      fprintf('Specific Entropy in KJ/KgK:%6.4f\n',SF);
      fprintf('Specific Heat at constant pressure ...
                  in kJ/(kgK):%10.4f\n',CPF);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVF);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUF);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      fprintf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
      disp(' ')
      return
end
if SVG>0
      disp(' ')
      fprintf('Specific Volume in m3/kg:%10.6f\n',SVG);
      fprintf('Internal Energy in kJ/Kg:%6.1f\n',UG);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      fprintf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUG);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      fprintf('Prandtl number:%6.3f\n',PRANDTLG);
      disp(' ')
      disp(' ')
      return
end
end
      disp('LIQUID ')
      disp(' ')
      fprintf('Specific Volume in m3/kg:%8.7f\n',SVF);
      fprintf('Internal Energy in kJ/kg:%6.1f\n',UF);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HF);
      fprintf('Specific Entropy in kJ/kgK:%6.4f\n',SF);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
      fprintf('Specific Heat atconstant volume in kJ/(kgK):%6.4f\n',CVF);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUF);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      fprintf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
      disp('VAPOR')
```

```
      disp(' ')
      fprintf('Specific Volume in m3/kg:%10.6f\n',SVG);
      fprintf('Internal Energy in kJ/kg:%6.1f\n',UG);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      fprintf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUG);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      fprintf('Prandtl number:%6.3f\n',PRANDTLG);
      disp(' ')
      disp(' ')
      disp(' ')
if IS==2
      fprintf('Surface tension in N/m:%7.6f\n',SIGMAFG);
else
      return
end
```

I.1.14 Function fp (αρχείο fp.m)

```
function[FP]=fp(ZZ)
%
%     COMPUTE PRESSURE OF LIGHT WATER IN MPa GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
transient;
qmust(ZZ);
%
%     CONVERT PRESSURE IN BARS TO PRESSURE IN MPa
%
PPP=PP/10.;
FP=ZZ*R*TT*(1+ZZ*Q+ZZ*ZZ*QDT)-PPP;
return
end
```

I.1.15 Function fpr3 (αρχείο fpr3.m)

```
function[FPR3]=fpr3(X3)
%
%     COMPUTE REDUCED PRESSURE OF LIGHT WATER VAPOR IN THERMODYNAMIC
%     SUB-REGIONS 3 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
%     INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
%
constants;
transient;
XSC0N=0.;
XSC1N=0.;
XSC2N=0.;
XSC3N=0.;
XSC4N=0.;
XSC6N=0.;
for  N=0:9
      if N>=2
            XSC0N=XSC0N+(1-N)*C(N)/(X3^N);
      if N>=2 & N<=6
            XSC1N=XSC1N+(1-N)*C(10+N)/(X3^N);
      end
      if N>=2 & N<=7
            XSC2N=XSC2N+(1-N)*C(20+N)/(X3^N);
      end
      if N>=2
            XSC3N=XSC3N+(1-N)*C(30+N)/(X3^N);
      end
      if N<=4
            XSC6N=XSC6N+C(60+N)/(TR^(N+2));
      end
end
XSC0N=XSC0N+(1-10)*C010/(X3^10);
XSC0N=XSC0N+(1-11)*C011/(X3^11);
XSC0N=XSC0N+C(1)+C012/X3;
XSC1N=XSC1N+C(11)+C(17)/X3;
XSC2N=XSC2N+C(21)+C(28)/X3;
XSC3N=XSC3N+C(31)+C310/X3;
XSC4N=5*C(41)*(TR-1)/((X3^6)*(TR^23));
XSC6N=6*(X3^5)*XSC6N;
FPR3=-XSC0N-XSC1N*(TR-1)-XSC2N*((TR-1)^2)-XSC3N*((TR-1)^3) ...
      +XSC4N-XSC6N-PR ;
return
end
```

I.1.16 Function fpr4 (αρχείο fpr4.m)

```
function[FPR4]=fpr4(X4)
%
%     COMPUTE REDUCED PRESSURE OF LIQUID LIGHT WATER IN THERMODYNAMIC
%     SUB-REGIONS 4 AND 5 CORRESPONDING TO REDUCED TEMPERATURE  AND
%     INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
%
constants;
transient;
Y=(1-TR)/(1-TR1);
XSDN1=0.D+0;
XSDN2=0.D+0;
XSDMN=0.D+0;
for  M=3:4
      for  N=1:5
            XSDN1=XSDN1+(N-1)*D(M,N)*(Y^M)/(X4^N);
      end
      XSDMN=XSDMN+XSDN1;
end
for  N=1:3
      XSDN2=XSDN2+(N-1)*D(5,N)*(X4^(N-2));
end
XSDN2=(Y^32)*XSDN2;
[FPR3]=fpr3(X4);
FPR4=FPR3+XSDMN-XSDN2;
return
end
```

I.1.17 Function fps (αρχείο fps.m)

```
function[FPS]=fps(TS)
%
%     COMPUTE SATURATION VAPOR PRESSURE OF LIGHT WATER IN MPa AS
%     FUNCTION OF REDUCED SATURATION TEMPERATURE
%
constants;
transient;
%
%     REDUCE SATURATION TEMPERATURE
%
      TR=TS/TC;
%
%     CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
%
      PPC=PC/10.;
      PPP=PP/10.;
      SUMKN=0.;
for N=1:5
      SUMKN=SUMKN+K(N)*((1-TR)^N);
end
      PRK=(1/TR)*SUMKN;
      PRK=PRK/(1+K(6)*(1-TR)+K(7)*((1-TR)^2));
      PRK=PRK-(1-TR)/(K(8)*((1-TR)^2)+K(9));
      FPS=exp(PRK)*PPC-PPP;
      return
end
```

I.1.18 Function fts (αρχείο fts.m)

```
function[FTS]=fts
%
%     COMPUTE FIRST APPROXIMATION OF LIGHT WATER SATURATION
%     TEMPERATURE IN K UNITS AS FUNCTION OF PRESSURE IN BARS
%     ACCORDING TO AUTHORS FIT.THIS APPROXIMATION IS TO BE USED
%     AS INITIAL GUESS FOR NUMERICAL CALCULATION OF ACTUAL
%     SATURATION TEMPERATURE OF LIGHT WATER
%
constants;
transient;
      A=219.88851D+0;
      B=146.70687D+0;
      C=0.19753D+0;
      FTS=A+B*(PP^C);
      return
end
```

I.1.19 Function ibm_jtc (αρχείο ibm_jtc.m)

```
function[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE ISOTHERMAL BULK MODULUS IN 1/MPa AND JOULE-THOMMSON
%     COEFFICIENT IN K/MPa
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
JTC=(1/(DD*CP))*((TT/DD)*(PTD/PDT)-1);
IBM=1./(DD*PDT);
return
end
```

I.1.20 Script initial.m

```
'INITIAL';
%
ZERO=0.D+0;
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
%     REGARDLESS OF PHASE
%
[D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC]=deal(0.);
[GAMMA A MU K PRANDTL SIGMA]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY
%     WATER SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
%
[DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF
JTCF]=deal(ZERO);
[GAMMAF AF MUF KF PRANDTLF]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY
%     WATER VAPOR SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
%
[DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG
JTCG]=deal(ZERO);
[GAMMAG AG MUG KG PRANDTLG]=deal(0.);
%
%     VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY
%     IN SATURATION STATES, SUFFIX FG STANDS FOR SATURATION
%
[HFG SIGMAFG LACFG]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR
%     THE MAIN DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS
%     OF DOUBLE SERIES EXPANSION OF Q
%
[Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2]=deal(0.);
```

I.1.21 Function light_wasp (αρχείο light_wasp.m)

```
function [P,T,TS,SVF,SVG,JR,IS] = light_wasp(JS,JP,P,T)
IER=0;
%
constants;
transient;
%
%     INITIALIZE VALUES OF DENSITIES AND SPECIFIC VOLUMES
%
[DF DG SVF SVG]=deal(0.);
%
%     INITIALIZE VALUES OF OTHER PROPERTIES
%
initial;
%
%     LIGHT WATER AND STEAM PROPERTY PACKAGE DEVELOPED BY
%     STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>
%
%     COMPUTE THERMODYNAMIC AND TRANSPORT PROPERTIES OF LIGHT WATER
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN DEGREES CENTIGRADE
%     MAINLY ACCORDING TO EQUATION OF STATE DEVELOPED BY HILL P.G.,
%     MACMILLAN R.D.C. AND LEE V.,AND EQUATIONS FOR TRANSPORT
%     PROPERTIES DEVELOPED BY MATSUNAGA N. AND NAGASHIMA A.
%     PARTS OF CODE STRUCTURE ARE BASED TO SIMILAR PROPERTY PACKAGE
%     FOR LIGHT WATER DEVELOPED BY HENDRICKS R.C.,PELLER I.C. AND
%     BARON A.K.PARTS OF EQUATION OF STATE FOR LIGHT WATER DEVELOPED
%     BY SCHMIDT E. AND GRIGULL U. ARE USED TO PROVIDE GOOD INITIAL
%     GUESSES FOR NUMERICAL CALCULATION OF HEAVY WATER DENSITY.
%     VALUE OF JS(INPUT) MUST BE 1.CALCULATED THERMODYNAMIC AND
%     TRANSPORT PROPERTIES ARE SPECIFIED BY JP(INPUT).JR(OUTPUT)
%     REPRESENTS THERMODYNAMIC SUB-REGION OF P-T DIAGRAMM IN WHICH
%     INPUT LIES.IS(OUTPUT) INDICATES RELATIVE POSITION OF INPUT TO
%     SATURATION CURVE.COMPUTE SATURATION PROPERTIES IF EITHER PRESSURE
%     OR TEMPERATURE IS EQUAL TO ZERO (FOR RESPECTIVE TEMPERATURE OR
%     PRESSURE)
ITEST=0;
%
%     CHECK FOR CORRECT CALL OF LIGHT_WASP ROUTINE
%
if JS~=1
      disp('Internal Error: Illegal JS parameter')
      return
end

if JP>63 | JP<0
      disp('Internal Error: Illegal JS parameter')
      return
end
PP=P;
TT=T;
TS=0.;
%
%     CHECK FOR OUT OF RANGE
%
[TS,JR,IS]=checkpt;
if JR==0
      return
end
P=PP;
```

```
T=TT;
JJ=JR;
%
%     DENSITY OF LIQUID HEAVY WATER
%
if JJ==1 | JJ==6 | JJ==4 | JJ==5
      [DF,IER]=densf;
      if DF~=0.
%
%     CONVERT DENSITY TO SPECIFIC VOLUME IN m3/Kg
%
      SVF=1/(1000*DF);
      DD=DF;
      end
%
%     CHECK FOR CRITICAL POINT VICINITY
%
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      ITEST=1;
      disp('Calculations near the critical point have no validity!')
      return
      end
%
%     OTHER PROPERTIES OF LIQUID HEAVY WATER
%
      if JP~=0
[B2F,B3F,B4F,DB2F,DB3F,DB4F,UF,PSIF,HF,SF,CPF,CVF,IBMF,JTCF,GAMMAF,..
.
AF,MUF,KF,PRANDTLF,SIGMAFG]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,...
PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
      end
end
%
%     DENSITY OF HEAVY WATER VAPOR
%
if JJ==2 | JJ==6 | JJ==3 | JJ==5
      [DG,IER]=densg;
      if DG~=0.
%
%     CONVERT DENSITY TO SPECIFIC VOLUME IN m3/kg
%
      SVG=1/(1000*DG);
      DD=DG;
      end
%
%     CHECK FOR CRITICAL POINT VICINITY
%
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      if ITEST==0
      disp('Calculations near the critical point have no validity!')
      return
      end
      end
%
%     OTHER PROPERTIES OF HEAVY WATER VAPOR
%
      if JP~=0
[B2G,B3G,B4G,DB2G,DB3G,DB4G,UG,PSIG,HG,SG,CPG,CVG,IBMG,JTCG,GAMMAG,..
.
AG,MUG,KG,PRANDTLG,SIGMAFG]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,...
PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
```

```
      end
end
if JJ==5 | JJ==6
%
%     COMPUTE LATENT HEAT OF HEAVY WATER IN kJ/kg
%
      HFG=HG-HF;
%
%     COMPUTE LAPLACE CONSTANT OF HEAVY WATER IN m
%
      [DG,IER]=densg;
      [DF,IER]=densf;
      if DF~=0. & DG~=0. & abs(DF-DG)>=1.D-3
      G=9.80665D+0;
      LACFG=sqrt(SIGMAFG/(G*1000*(DF-DG)));
      end
end
disp(' ')
disp(' ')
disp(' ')
return
end
```

I.1.22 Function newton (αρχείο newton.m)

```
function[X,IER]=newton(f,df,EPS,NDEC,X,ITMAX)
%     FIND REAL ZERO X OF REAL FUNCTION F GIVEN GOOD INITIAL GUESS
%     X AND FUNCTION DF (1st PARTIAL DERIVATIVE OF F BY X).ROOT X
%     IS ACCEPTED IF ABSOLUTE VALUE OF F(X).LE.EPS(INPUT).ROOT X
%     IS ACCEPTED IF TWO SUCCESSIVE APPROXIMATIONS AGREE TO FIRST
%     NDEC(INPUT) DECIMAL DIGITS.X(INPUT) IS AN INITIAL
%     VALUE.X(OUTPUT) CONTAINS COMPUTED ROOT.ITMAX(INPUT) IS MAXIMUM
%     ALLOWBLE NUMBER OF NEWTON_RAPHSON ITERATIONS USED ON
%     ROOT.IER(OUTPUT) WARNING ERROR PARAMETER=N,N=1 ROOT X WAS
%     BYPASSED BECAUSE DF BECOMES TOO SMALL.X IS SET TO 111111.THIS
%     ERROR CONDITION MAY CAUSE AN OVERFLOW,N=2 ROOT X WAS BYPASSED
%     BECAUSE ITMAX WAS EXCEEDED. X IS SET TO 222222,N=3 SEVERAL OF
%     THE ABOVE ERROR CONDITIONS OCCURED.X IS SET TO EITHER 111111.
%     OR 222222. AS ABOVE.DOUBLE PRECISION ROUTINE.
%
TOL=10.^(-NDEC); IER=0;
if ITMAX>100
      ITMAX=100;
end
for  K=1:ITMAX
      dd=feval(df,X);
      if abs(dd)<EPS
      X=111111.D+0;
      IER=1;
      return
      end
      ff=feval(f,X);
      dd=feval(df,X);
      XK=X-ff/dd;
      if K==41
      TOL=TOL*10;
      end
      if K==61
            TOL=TOL*10;
      end
      if K==81
      TOL=TOL*10 ;
      end
      if abs(XK-X)<TOL
      X=XK;
      return
      else
      X=XK;
      ff=feval(f,X);
      if abs(ff)<EPS
      return
      end
      end
end
      X=222222.D+0;
      if IER==0
      IER=2;
      else
      IER=3;
      end
return
end
```

I.1.23 Function prl (αρχείο prl.m)

```
function[PRL]=prl
%
%     COMPUTE REDUCED PRESSURE OF LIGHT WATER CORRESPONDING TO
%     REDUCED TEMPERATURE OF HEAVY WATER CALCULATED FROM GIVEN
%     TEMPERATURE IN K UNITS ALONG THE BOUNDARY BETWEEN THERMODYNAMIC
%     SUB-REGIONS 2 AND 3
%
constants;
transient;
L=7.160997524D+0;
PRL=((TR2-TR)*PR1+(TR-TR1)*PR2-L*(TR2-TR)*(TR-TR1))/(TR2-TR1);
return
end
```

I.1.24 Function prs (αρχείο prs.m)

```
function[PRS]=prs
%
%     COMPUTE SATURATION VAPOR PRESSURE OF LIGHT WATER IN MPa AS
%     FUNCTION OF REDUCED SATURATION TEMPERATURE
%
constants;
transient;
%
%     CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
%
      PPC=PC/10.;
      PPP=PP/10.;
      SUMKN=0.D+0;
for N=1:5
      SUMKN=SUMKN+K(N)*((1-TR)^N);
end
      PRK=(1/TR)*SUMKN;
      PRK=PRK/(1+K(6)*(1-TR)+K(7)*((1-TR)^2));
      PRK=PRK-(1-TR)/(K(8)*((1-TR)^2)+K(9));
      PRS=exp(PRK);
      return
end
```

I.1.25 Function qmust (αρχείο qmust.m)

```
function[]=qmust(ZZ)
constants;
transient;
%
%     PRIMARY CONSTANTS OF DOUBLE SERIES EXPANSION OF Q
%
%     COEFFICIENTS Aij
%
A=[29.492937D+0,-132.13917D+0,274.64632D+0,...
-360.93828D+0,342.18431D+0,-244.50042D+0,155.18535D+0,...
5.9728487D+0,-410.30848D+0,-416.05860D+0;...
-5.1985860D+0,7.7779182D+0,-33.301902D+0,-16.254622D+0,...
-177.31074D+0,127.48742D+0,137.46153D+0,155.97386D+0,...
337.31180D+0,-209.88866;6.8335354D+0,-26.149751D+0,...
65.326396D+0,-26.181978D+0,0.0D+0,0.0D+0,0.0D+0,...
0.0D+0,-137.46618D+0,-733.96848D+0;-0.1564104D+0,...
-0.72546108D+0,9.2734289D+0,4.3125840D+0,0.0D+0,...
0.0D+0,0.0D+0,0.0D+0,6.7874983D+0,10.401717D+0;...
-6.3972405D+0,26.409282D+0,-47.740374D+0,...
56.323130D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0, ...
136.87317D+0,645.81880D+0;-3.9661401D+0,...
15.453061D+0,-29.142470D+0,29.568796D+0,...
0.0D+0,0.0D+0,0.0D+0,0.0D+0,79.847970D+0,...
399.17570D+0;-0.69048554D+0,...
2.7407416D+0,-5.1028070D+0,3.9636085D+0,...
0.0D+0,0.0D+0,0.0D+0,0.0D+0,13.041253D+0,71.531353D+0];
A=transpose(A);
%
%     COEFFICIENTS Ci
%
      C=[1855.3865D+0,3278.642D+0,-379.03D+0,46.174D+0,-1021.17];
%
%     OTHER CONSTANTS
%
      TAJ1=1.544912D+0;
      TAJ2=2.5D+0;
      E=4.8D+0;
      RAJ1=0.634D+0;
      RAJ2=1.0D+0;
%
%     TEMPERATURE PARAMETER
%
      TAU=1000/TT;
      TAUC=TAJ1;
      EX=exp(-E*ZZ);
      SUMI1=0.;
      SUMI3=0.;
      SUMI5=0.;
for I=1:8
      SUMI1=SUMI1+A(I,1)*(ZZ-RAJ1)^(I-1);
      if I>=2
            SUMI3=SUMI3+(I-1)*A(I,1)*(ZZ-RAJ1)^(I-2);
      end
      if I>=3
            SUMI5=SUMI5+A(I,1)*(I-1)*(I-2)*(ZZ-RAJ1)^(I-3);
      end
end
      SUMJ1=0.;
```

```
      SUMJ2=0.;
      SUMJ3=0.;
      SUMJ4=0.;
      SUMJ5=0.;
      SUMJ6=0.;
for J=1:7
      SUMI2=0.;
      SUMI4=0.;
      SUMI6=0.;
for I=1:8
      SUMI2=SUMI2+A(I,J)*(ZZ-RAJ2)^(I-1);
      if I>=2
            SUMI4=SUMI4+(I-1)*A(I,J)*(ZZ-RAJ2)^(I-2);
      end
      if I>=3
            SUMI6=SUMI6+A(I,J)*(I-1)*(I-2)*(ZZ-RAJ2)^(I-3);
      end
end
      SUMI2=SUMI2+EX*(A(9,J)+A(10,J)*ZZ);
      SUMI4=SUMI4+EX*(-E*(A(9,J)+A(10,J)*ZZ)+A(10,J));
      SUMI6=SUMI6+EX*(-E)*(-E*A(9,J)+A(10,J)*(2-E*ZZ));
      if J>=2
            SUMJ1=SUMJ1+((TAU-TAJ2)^(J-2))*SUMI2;
      end
      if J>=3
            SUMJ2=SUMJ2+(J-2)*((TAU-TAJ2)^(J-3))*SUMI2;
      end
      if J>=2
            SUMJ3=SUMJ3+((TAU-TAJ2)^(J-2))*SUMI4;
      end
      if J>=3
            SUMJ4=SUMJ4+(J-2)*((TAU-TAJ2)^(J-3))*SUMI4;
      end
      if J>=2
            SUMJ5=SUMJ5+((TAU-TAJ2)^(J-2))*SUMI6;
      end
      if J>=4
            SUMJ6=SUMJ6+(J-2)*(J-3)*((TAU-TAJ2)^(J-4))*SUMI2;
      end
end
      SUMI1=SUMI1+EX*(A(9,1)+A(10,1)*ZZ);
      SUMI3=SUMI3+EX*(-E*(A(9,1)+A(10,1)*ZZ)+A(10,1));
      SUMI5=SUMI5+EX*(-E)*(-E*A(9,1)+A(10,1)*(2-E*ZZ));
%
%     DATA-FITTING FUNCTION,Q
%
      Q=SUMI1+(TAU-TAUC)*SUMJ1;
%
%     1st PARTIAL DERIVATIVE OF Q BY TEMPERATURE
%
      QTD=SUMJ1+(TAU-TAUC)*SUMJ2;
%
%     1st PARTIAL DERIVATIVE OF Q BY DENSITY
%
      QDT=SUMI3+(TAU-TAUC)*SUMJ3;
%
%     2nd PARTIAL DERIVATIVE OF Q BY DENSITY AND TEMPERATURE
%
      Q2DT=SUMJ3+(TAU-TAUC)*SUMJ4;
%
%     2nd PARTIAL DERIVATIVE OF Q BY DENSITY
```

```
%
      Q2D2T=SUMI5+(TAU-TAUC)*SUMJ5;
%
%     2nd PARTIAL DERIVATIVE OF Q BY TEMPERATURE
%
      Q2T2D=2*SUMJ2+(TAU-TAUC)*SUMJ6;
      SUMI7=0.;
      SUMI8=0.;
      SUMI9=0.;
for I=1:3
      SUMI7=SUMI7+C(I)*((TT/1000.)^(I-1));
      SUMI8=SUMI8+C(I)*(I-1)*((TT/1000.)^(I-2));
      SUMI9=SUMI9+C(I)*(I-1)*(I-2)*((TT/1000.)^(I-3));
end
%
%     REFERENCE FUNCTION,PSI0
%
      PSI0=SUMI7+C(4)*log(TT)+C(5)*TT*log(TT)/1000;
%
%     1st PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
%
      PSI0T=SUMI8/1000+C(4)/TT+C(5)*log(TT)/1000+C(5)/1000;
%
%     2nd PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
%
      PSI02T2=SUMI9*(1.D-6)-C(4)/(TT*TT)+C(5)/(1000*TT);
      return
end
```

I.1.26 Function shp (αρχείο shp.m)

```
function[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE SPECIFIC HEAT OF LIGHT WATER AT CONSTANT PRESSURE IN
%     kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
      PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
return
end
```

I.1.27 Function shr_sove (αρχείο shr_sove.m)

```
function[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE ISENTROPIC EXPANSION COEFFICIENT(SPECIFIC HEAT RATIO)
%     AND SONIC VELOCITY IN m/s OF HEAVY WATER
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)*Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
%
%     COMPUTE CP
%
CP=HTD-HDT*PTD/PDT;
%
%     COMPUTE CV
%
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
GAMMA=CP/CV;
A=(10^(3/2.))*(1./sqrt(1/PDT-(TT/(CP*(DD^2)))*(PTD^2)/(PDT^2)));
return
end
```

I.1.28 Function shv (αρχείο shv.m)

```
function[CV]=shv(TT,DD,PSI02T2,Q2T2D)
%
%     COMPUTE SPECIFIC HEAT OF LIGHT WATER AT CONSTANT VOLUME IN
%     kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
return
end
```

I.1.29 Function svlwl (αρχείο svlwl.m)

```
function[DL,IER]=svlwl
%
%     COMPUTE DENSITY OF LIQUID LIGHT WATER IN g/cm3 CORRESPONDING TO
%     REDUCED PRESSURE AND TEMPERATURE OF LIGHT WATER CALCULATED FROM
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
%
IER=0;
constants;
transient;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS IN THERMODYNAMIC
%     SUB-REGION 1
%
AM=[8.438375405D-1 5.362162162D-4 1.72D+0 7.342278489D-2 ...
      4.97585887D-2 6.5371543D-1 1.15D-6 1.5108D-5 ...
      1.4188D-1 7.002753165D+0 2.995284926D-4 2.04D-1];
A=[0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      7.982692717D+0 -2.616571843D-2 1.52241179D-3 2.284279054D-2 ...
      2.421647003D+2 1.269716088D-10 2.074838328D-7 2.17402035D-8 ...
      1.105710498D-9 1.293441934D+1 1.308119072D-5 6.047626338D-14];
      EPS=1.D-5; NDEC=5; ITMAX=1000;
%
%     ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
if JJ==6 | JJ==5
      [PRS]=prs;
      PR=PRS;
end
Y=1-(AM(1)*TR*TR)-(AM(2)/(TR^6));
Z=AM(3)*Y*Y-2*AM(4)*TR+2*AM(5)*PR;
if Z<=0
      Z=Y;
else
      Z=Y+sqrt(Z);
end
      X1=A(11)*AM(5)/(Z^(5./17.))...
      +(A(12)+A(13)*TR+A(14)*TR*TR+A(15)*((AM(6)-TR)^10)...
      +A(16)/(AM(7)+(TR^19)))...
       (A(17)+2*A(18)*PR+3*A(19)*PR*PR)/(AM(8)+(TR^11))...
      -A(20)*(TR^18)*(AM(9)+TR*TR)...
      *(-3/((AM(10)+PR)^4)+AM(11))...
      +3*A(21)*(AM(12)-TR)*PR*PR+4*(A(22)/(TR^20))*(PR^3);
if JJ==4
      X4=X1;
      f='fpr4';
      df='dfpr4';
      [X4,IER]=newton(f,df,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
if JJ==5
      [DV,IER]=svlwv;
      X4=1./(DV*VC);
      X4=X1;
      f='fpr4';
      df='dfpr4';
      [X4,IER]=newton(f,df,EPS,NDEC,X4,ITMAX);
      if X4<=0
      X4=10*R*TT/(PP*VC);
```

```
      f='fpr4';
      df='dfpr4';
      [X4,IER]=newton(f,df,EPS,NDEC,X4,ITMAX);
      end
      X1=X4;
end
PR=PP/PC;
VL=X1*VC;
DL=1/VL;
return
end
```

I.1.30 Function svlwv (αρχείο svlwv.m)

```
function[DV,IER]=svlwv
%
%     COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 CORRESPONDING TO
%     REDUCED PRESSURE AND TEMPERATURE OF LIGHT WATER CALCULATED FROM
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
%
IER=0;
constants;
transient;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS
%
%     THERMODYNAMIC SUB-REGION 2
%
B0=7.633333333D-1; B90=1.936587558D+2;
B=[6.670375918D-2,8.390104328D-2,4.520918904D-1,...
-5.975336707D-1,5.958051609D-1,1.190610271D-1,...
1.683998803D-1,6.552390126D-3,-1.388522425D+3;...
1.388983801D+0,2.614670893D-2,1.069036614D-1,...
-8.847535804D-2,-5.159303373D-1,-9.867174132D-2,...
-5.809438001D-2,5.710218649D-4,4.126607219D+3;...
0.D+0,-3.373439453D-2,0.D+0,0.D+0,2.075021122D-1,0.D+0,...
0.D+0,0.D+0,-6.508211677D+3;0.D+0,...
0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,5.745984054D+3;...
0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,-2.693088365D+3;...
0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,5.235718623D+2];
B=transpose(B);

      BM(6,1)=4.006073948D-1; BM(7,1)=8.636081627D-2;
      BM(8,1)=-8.532322921D-1; BM(8,2)=3.460208861D-1;

%
%     NUMBERS OF TERMS n(m) AND l(m),AND EXPONENTS z(m,n) AND x(m,n)
%
      NM=[2,3,2,2,3,2,2,2];
      ZM=[13,18,18,25,32,12,24,24;...
            3,2,10,14,28,11,18,14;...
            0,1,0,0,24,0,0,0];
      ZM=transpose(ZM);
      LM=[0,0,0,0,0,1,1,2];
      XM=[0,0,0,0,0,14,19,54;
            0,0,0,0,0,0,0,27];
      XM=transpose(XM);
      EPS=1.D-5;
      NDEC=5;
      ITMAX=100;
%
%     ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
if JJ==6 | JJ==5
      [PRS]=prs;
      PR=PRS;
end
I1=10*R*TC/(PC*VC);
X=exp(B0*(1-TR));
VSM1=0.D+0;
VSM2=0.D+0;
VSN=0.D+0;
```

```
for M=1:8
      VSN1=0.D+0;
      VSN2=0.D+0;
      VSL=0.D+0;
      if M<=5
      for N=1:NM(M)
            VSN1=VSN1+B(M,N)*(X^ZM(M,N));
      end
            VSM1=VSM1+M*(PR^(M-1))*VSN1;
      else
      for N=1:NM(M)
            VSN2=VSN2+B(M,N)*(X^ZM(M,N));
      end
      for L=1:LM(M)
            VSL=VSL+BM(M,L)*(X^XM(M,L));
      end
            VSM2=VSM2+((M-2)*(PR^(1-M))*VSN2)/(((PR^(2-M))+VSL)^2);
      end
end
[PRL]=prl;
VSN=VSN+(11.*(PR/PRL)^10)*B90;
for N=1:6
      [PRL]=prl;
      VSN=VSN+(11.*(PR/PRL)^10)*B(9,N)*(X^N);
end
X2=I1*(TR/PR)-VSM1-VSM2+VSN;
if JJ==3 | JJ==5
      if X2<=0
            X3=I1*(TR/PR);
      else
      X3=X2;
      end
      F='fpr3';
      DF='dfpr3';
      [X3,IER]=newton(F,DF,EPS,NDEC,X3,ITMAX);
      X2=X3;
end
PR=PP/PC;
VV=VC*X2;
DV=1/VV;
return
end
```

I.1.31 Function tension (αρχείο tension.m)

```
function[SIGMA]=tension(TT,JJ)
%
%     COMPUTE SURFACE TENSION OF LIGHT WATER IN N/m GIVEN
%     TEMPERATURE IN K UNITS ACCORDING TO EQUATION DEVELOPED BY
%     STRAUB J.,ROSNER N. AND GRIGULL U.
%
constants;
      SIGMA0=235.8D-3; M=1.256D+0; B=-0.625D+0;
      TCSTR=647.15D+0;
      SIGMA=0.;
      if JJ~=6 & JJ~=5
      return
      end
      TAUT=1-TT/TCSTR;
      SIGMA=SIGMA0*(TAUT^M)*(1+B*TAUT);
      return
end
```

I.1.32 Function total (αρχείο total.m)

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
A,MU,K,PRANDTL,SIGMA]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
%     IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
%     MUST BE CALLED TWICE.
%
%     ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
%     SPECIFICATION,JP
%
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 ...
      42 43 46 47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44...
       45 46 47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44...
       45 46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51...
       52 53 54 55 56 57 58 59 60 61 62 63];
[D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC]=deal(0.);
[GAMMA A MU K PRANDTL SIGMA]=deal(0.);
if mod(JP,2)< 0
      T60;
end
if mod(JP,2)==0
      T70;
      T100;
end
if mod(JP,2)>0
      T60;
end
%
%     VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
%     TEMPERATURE AND INTERNAL ENERGY
%
function[]=T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70;
return
end
function[]=T70
for I=1:32
      if JP-JPC1(I)<0
      T100;
      return
      end
      if JP-JPC1(I)==0
      T90;
      return
      end
end
return
end
%
%     ENTHALPY AND ENTROPY
%
function[]=T90
```

```
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
T100;
return
end
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120;
      return
      end
end
return
end
%
%     SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT
%     VOLUME, SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
%     MODULUS AND JOULE-THOMSON COEFFICIENT
%
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130;
return
end
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150;
      return
      end
end
return
end
%
%     VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
%
function[]=T150
[MU]=viscosity(TT,DD);
[K]=conductive(TT,DD);
if K~=0.D+0
      PRANDTL=1000.*MU*CP/K;
end
T160;
return
end
function[]=T160
for I=1:32
      if JP-JPC4(I)<0
      T190;
      return
```

```
      end
      if JP-JPC4(I)==0
      T180;
      return
      end
end
return
end
%
%     surface tension
%
function[]=T180
if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
end
T190;
return
end
function[]=T190
      if JP-32<0
      T200;
      return
      end
      if JP-32==0
      T210;
      return
      end
      if JP-32>0
      T210;
      return
      end
return
end
      function[]=T200
      return
      end
      function[]=T210
      return
      end
end
```

I.1.33 Script transient.m

```
'TRANSIENT';
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
%
global PP PR TT TR DD JJ

global D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC
global GAMMA A MU K PRANDTL
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY
%     WATER SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
%
global DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF JTCF
global GAMMAF AF MUF KF PRANDTLF
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
%     VAPOR SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
%
global DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG JTCG
global GAMMAG AG MUG KG PRANDTLG
%
%     VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN
%     SATURATION STATES,SUFFIX FG STANDS FOR SATURATION
%
global HFG SIGMAFG LACFG
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE
%     MAIN DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE
%     SERIES EXPANSION OF Q
global Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2
```

I.1.34 Function viscosity (αρχείο viscosity.m)

```
function[MU]=viscosity(TT,DD)
%
%     COMPUTE VISCOSITY OF LIGHT WATER IN kg/(ms) GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%     FROM AN EQUATION SUITABLE FOR INDUSTRIAL USE
%
constants;
H=[1.00000D+0,0.978197D+0,0.579829D+0,-0.202354D+0];
HH=[0.5132047D+0,0.3205656D+0,0.D+0,0.D+0,...
      -0.7782567D+0,0.1885447D+0;0.2151778D+0,...
      0.7317883D+0,1.241044D+0,1.476783D+0,0.D+0,0.D+0;...
-0.2818107D+0,-1.070786D+0,-1.263184D+0,0.D+0,0.D+0, 0.D+0;...
0.1778064D+0,0.4605040D+0,0.2340379D+0,...
-0.4924179D+0,0.D+0,0.D+0;-0.04176610D+0,0.D+0,...
0.D+0,0.1600435D+0,0.D+0,0.D+0;...
0.D+0,-0.01578386D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
0.D+0,0.D+0,0.D+0,-0.003629481D+0,0.D+0,0.D+0];
HH=transpose(HH);
      NC=55.071D-6;
      PSEUDODR=DD/PSEUDODC;
      PSEUDOTR=TT/PSEUDOTC;
      NR0=0.D+0;
for I=1:4
      NR0=NR0+H(I)/(PSEUDOTR^(I-1));
end
      NR0=sqrt(PSEUDOTR)/NR0;
      NR1=0.D+0;
for J=1:6
for K=1:7
      NR1=NR1+HH(J,K)*(((1./PSEUDOTR)-1)^(J-1))*((PSEUDODR-1)^(K-1));
end
end
      NR1=exp(PSEUDODR*NR1);
      NR2=1.D+0;
      NR=NR0*NR1*NR2;
      MU=NR*NC;
      return
end
```

I.2 Περιβάλλον προγραμματισμού SCILAB

Τα υποπρογράμματα παρατίθενται κατά την αλφαβητική σειρά της ονομασίας τους. Συμπεριλαμβάνεται το κυρίως πρόγραμμα example.sci.

I.2.1 Function checkpt (αρχείο checkpt.sci)

```
function[TS,JR,IS]=checkpt
IER=0;
TS=0.;
JR=0;
IS=0;
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
EPS=1.D-5; NDEC=5; ITMAX=100;
//
//    CHECK PRESSURE FOR OUT OF RANGE
//
if (PP<PMIN | PP>PMAX) & PP~=0.
      disp(' ')
      disp('Desired Pressure out of range')
      disp(' ')
      return
end
//
//    CHECK TEMPERATURE FOR OUT OF RANGE
//
if (TT<TMIN | TT>TMAX) & (TT~=0.)
      disp(' ')
      disp('Desired Temperature out of range')
      disp(' ')
      return
end
//
//    IF PRESSURE AND TEMPERATURE ARE EQUAL TO 0 CHECK FOR CORRECT
//    CALL OF SATURATION PROPERTIES
//
if PP==0. & TT==0.
      disp(' ')
      disp('Both pressure and temperature cannot be 0.!')
      disp(' ')
      return
end
//
//    IF PRESSURE OR TEMPERATURE IS EQUAL TO 0 CHECK FOR OUT OF
//    SATURATION RANGE
//
if (PP>PC & TT==0.) | (TT>TC & PP==0.)
      disp(' ')
      disp('Such a saturation state does not exist!')
      disp(' ')
      return
end
if TT==0.;
      TT=TT+273.15D+0;
end
if PP==0.;
      [FPS]=fps(TT);
      PP=FPS*10;
```

```
      TS=TT;
      IS=1;
else
if PP<=PC & TT<=TC
      [FTS]=fts();
      TS=FTS;
      [FPS]=fps(TS);
      if abs(FPS)>EPS
            [TS,IER]=newton(fps,dfps,EPS,NDEC,TS,ITMAX);
            IS=1;
      end
      if TT==273.15D+0
            TT=TS;
      end
      end
end
if abs(TT-TS)<=1.D+0
      IS=2;
end
//
//    REDUCED PRESSURE AND TEMPERATURE
//
PR=PP/PC;
TR=TT/TC;
//
//    DETERMINE THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
if TR>=TRT & TR<=TR1
      if abs(TT-TS)<=1.D-5
            JR=6;
            return
      else
            [PRS]=prs();
            if PR>=0. & PR<PRS
            JR=2;
            return
            end
            if PR>PRS & PR<=PR2
            JR=1;
            return
            end
      end
end
if TR>=TR &((TR-1)<1.D-5)
      if abs(TT-TS)<=1.D-5
            JR=5;
            return
      end
else
      [PRL]=prl();
      [PRS]=prs();
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<PRS
            JR=3;
            return
      end
      [PRS]=prs();
      if PR>PRS & PR<PR2
```

```
            JR=4;
            return
      end
end
if (abs(TR-1))>=1.D-5 & TR<TR2
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<=PR2
            JR=3;
            return
      end
end
if TR>=TR2 & TR<=TR3
      JR=2;
      return
end
if JR==0
      disp(' ')
      disp('Internal Error: JR indicator value is zero')
      disp(' ')
      return
end
return
endfunction
```

I.2.2 Function conductive (αρχείο conductive.sci)

```
function [KK]=conductive(TT,DD)
//
//    COMPUTE THERMAL CONDUCTIVITY OF LIGHT WATER IN W/(mK) GIVEN
//    TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//    FROM AN EQUATION SUITABLE FOR INDUSTRIAL USE
//
exec('F:\scilab_light_water\transient.sci');
exec('F:\scilab_light_water\constants.sci');
      A=[0.0102811D+0 0.0299621D+0 0.0156146D+0 -0.00422464D+0];
      B=[-0.397070D+0 0.400302D+0 1.060000D+0];
      B1=-0.171587D+0;
      B2=2.392190D+0;

      C1= 0.642857D+0;
      C2=-4.11717D+0;
      C3=-6.17937D+0;
      C4=0.00308976D+0;
      C5=0.0822994D+0;
      C6=10.0932D+0;

      D1=0.0701309D+0;
      D2=0.0118520D+0;
      D3=0.00169937D+0;
      D4=-1.0200D+0;
      ELC=1.D+0;
      PSEUDODR=DD/PSEUDODC;
      PSEUDOTR=TT/PSEUDOTC;
      ELR0=0.D+0;
for I=1:4
      ELR0=ELR0+A(I)*(PSEUDOTR^(I-1));
end
      ELR0=sqrt(PSEUDOTR)*ELR0;
      ELR1=B(1)+B(2)*PSEUDODR+B(3)*exp(B1*((PSEUDODR+B2)^2));
      DELTATR=abs(PSEUDOTR-1.)+C4;
      Q=2.+C5/(DELTATR^0.6);
      R=Q+1.;
      SS=1./DELTATR;
if PSEUDOTR<1.D+0
      SS=C6/(DELTATR^0.6);
end
ELR2=(D1/(PSEUDOTR^10)+D2)*(PSEUDODR^1.8)*exp(C1*(1-(PSEUDODR^2.8)))...
      +D3*SS*(PSEUDODR^Q)*exp((Q/R)* ...
      (1-(PSEUDODR^R)))+D4*exp(C2*(PSEUDOTR^1.5)+C3/(PSEUDODR^5));
      EL=ELC*(ELR0+ELR1+ELR2);
      KK=EL;
      return
endfunction
```

I.2.3 Script constants.sci

```
"constants";
//
//    SCRIPT CONTAINING ALL CONSTANTS OF LIGHT_WASP
//    BEING USED AT LEAST TWICE.ALL OTHER ONCE USED CONSTANTS ARE
//    PART OF CORRESPONDING ASSIGNMENTS IN INDIVIDUAL SUBPROGRAMS.
//
global PMIN PMAX TMIN TMAX PC TC VC TRT TR1 TR2 TR3 PR1 PR2 R K C
global C010
global C011 C012 C310 D PSEUDODC PSEUDOTC PSEUDOPC JPC1 JPC2 JPC3
global JPC4

//    COMMONS LABELLED WITH Z REPRESENT ALL CONSTANTS OF LIGHT_WASP
//    BEING USED AT LEAST TWICE
//
//
//    PRESSURE AND TEMPERATURE RANGE OF EQUATIONS FOR THERMODYNAMIC
//    AND TRANPORT PROPERTIES OF LIGHT WATER
//
      PMIN=0.006D+0;  PMAX=1000.D+0; TMIN=273.16D+0; TMAX=1273.15D+0;
//
//    CRITICAL PRESSURE,TEMPERATURE AND SPECIFIC VOLUME OF LIGHT
//    WATER
//
      PC=221.2D+0; TC=647.3D+0; VC=3.17D+0;
//
//    REDUCED TEMPERATURE RANGES OF THERMODYNAMIC SUB-REGIONS
//
      TRT=4.21999073D-1; TR1=9.626911787D-1; TR2=1.333462073D+0;
      TR3=1.657886606D+0;
//
//    MAXIMUM REDUCED PRESSURE
//
      PR1=7.475191707D-1; PR2=4.520795660D+0;
//
//    LIGHT WATER CONSTANT
//
      R=0.46151D+0;
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS OF EQUATION OF STATE
//    OF LIGHT WATER BEING USED IN THIS PACKAGE
//
//
//    SATURATION LINE
//
K=[-7.691234564D+0 -2.608023696D+1 -1.681706546D+2 6.423285504D+1 ...
      -1.189646225D+2 4.167117320D+0 2.097506760D+1 1.D+9 6.D+0];
//
//    THERMODYNAMIC SUB-REGIONS 3 AND 4
//
C=[-1.72260420D-2 -7.77175039D+0 4.20460752D+0 -2.76807038D+0 ...
      2.10419707D+0 -1.14649588D+0 2.23138085D-1 1.16250363D-1 ...
      -8.20900544D-2 0.D+0,7.08636085D-1 1.23679455D+1 ...
      -1.20389004D+1 5.40437422D+0 -9.93865043D-1 6.27523182D-2 ...
      -7.74743016D+0 0.D+0 0.D+0 0.D+0 -4.29885092D+0 ...
      4.31430538D+1 -1.41619313D+1 4.04172459D+0 ...
      1.55546326D+0 -1.66568935D+0 3.24881158D-1 2.93655325D+1 ...
      0.D+0 0.D+0 7.94841842D-6 8.0885947D+1 -8.36153380D+1 ...
      3.58636517D+1 7.51895954D+0 -1.26160640D+1 1.09717462D+0 ...
      2.12145492D+0 -5.46529566D-1 2.75971776D-6 -5.09073985D-4 ...
```

```
      0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      2.10636332D+2 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      0.D+0 0.D+0 5.528935335D-2 -2.336365955D-1 3.697071420D-1 ...
      -2.596415470D-1 6.828087013D-2];

      C010=1.94129239D-2;
      C011=-1.69470576D-3;
      C012=-4.311577033D+0;
      C310=8.32875413D+0;

//
//    THERMODYNAMIC SUB-REGION 4
//
D=[0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
      0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
      -1.717616747D+0,3.526389875D+0,-2.690899373D+0,...
      9.070982605D-1, -1.138791156D-1;1.301023613D+0,...
      -2.642777743D+0,1.996765362D+0,-6.661557013D-1,...
      8.270860589D-2;3.426663535D-4,-1.236521258D-3,...
      1.155018309D-3,0.D+0,0.D+0];


//    PSEUDOCRITICAL DENSITY AND TEMPERATURE OF LIGHT WATER TO BE
//    USED IN THE CALCULATION OF THERMAL CONDUCTIVITY AND VISCOCITY
//
      PSEUDODC=0.317363D+0; PSEUDOTC=647.27D+0; PSEUDOPC=221.15D+0;
```

I.2.4 Function densf (αρχείο densf.sci)

```
function[DF,IER]=densf
//
//    COMPUTE DENSITY OF LIQUID HEAVY WATER IN g/cm3 GIVEN PRESSURE
//    IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
//    IS EQUAL TO 0 SATURATED LIQUID DENSITY IS RETURNED
//
IER=0;
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
EPS=1.D-5; NDEC=5; ITMAX=100;
DF=0; DL=0;
[DL,IER]=svlwl();
[DF,IER]=newton(fp,dfpd,EPS,NDEC,DL,ITMAX);
ZZ=DF;
qmust(ZZ);
return
endfunction
```

I.2.5 Function densg (αρχείο densg.sci)

```
function[DG,IER]=densg
//
//    COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 GIVEN PRESSURE
//    IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
//    IS EQUAL TO 0 SATURATED VAPOR DENSITY IS RETURNED
//
IER=0;
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
EPS=1.D-5; NDEC=5;ITMAX=100;
DG=0; DV=0;
[DV, IER]=svlwv();
[DG, IER]=newton(fp,dfpd,EPS,NDEC,DV,ITMAX);
DD=DG;
ZZ=DG;
qmust(ZZ);
return
endfunction
```

I.2.6 Function dfpd (αρχείο dfpd.sci)

```
function[DFPD]=dfpd(ZZ)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF LIGHT WATER PRESSURE BY
//    DENSITY
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
DFPD=R*TT*(1+2*ZZ*Q+4*ZZ*ZZ*QDT+ZZ*ZZ*ZZ*Q2D2T);
return
endfunction
```

I.2.7 Function dfpr3 (αρχείο dfpr3.sci)

```
function[DFPR3]=dfpr3(X3)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIGHT
//    WATER VAPOR BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
//    SUB-REGIONS 3 AND 5
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
DXSC0N=0.;
DXSC1N=0.;
DXSC2N=0.;
DXSC3N=0.;
DXSC4N=0.;
DXSC6N=0.;
for N=0:9
      if N>=2
            DXSC0N=DXSC0N+(N^2-N)*C(N)/(X3^(N+1));
      end
      if N>=2 & N<=6
            DXSC1N=DXSC1N+(N^2-N)*C(10+N)/(X3^(N+1));
      end
      if N>=2 & N<=7
            DXSC2N=DXSC2N+(N^2-N)*C(20+N)/(X3^(N+1));
      end
      if N>=2
            DXSC3N=DXSC3N+(N^2-N)*C(30+N)/(X3^(N+1));
      end
      if N<=4
            DXSC6N=DXSC6N+C(60+N)/(TR^(N+2));
      end
end
DXSC0N=DXSC0N+(100-10)*C010/(X3^11);
DXSC0N=DXSC0N+(121-11)*C011/(X3^12);
DXSC0N=DXSC0N-C012/(X3^2);
DXSC1N=DXSC1N-C(17)/(X3^2);
DXSC2N=DXSC2N-C(28)/(X3^2);
DXSC3N=DXSC3N-C310/(X3^2);
DXSC4N=-30*C(41)*(TR-1)/((X3^7)*(TR^23));
DXSC6N=30*(X3^4)*DXSC6N;
DFPR3=-DXSC0N-DXSC1N*(TR-1)-DXSC2N*((TR-1)^2)...
      -DXSC3N*((TR-1)^3)+DXSC4N-DXSC6N;
return
endfunction
```

I.2.8 Function dfpr4 (αρχείο dfpr4.sci)

```
function[DFPR4]=dfpr4(X4)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIQUID
//    LIGHT WATER BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
//    SUB-REGIONS 4 AND 5
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
Y=(1-TR)/(1-TR1);
DXSDN1=0.;
DXSDN2=0.;
DXSDMN=0.;
for M=3:4
      for N=1:5
            DXSDN1=DXSDN1-N*(N-1)*D(M,N)*(Y^M)/(X4^(N+1));
      end
      DXSDMN=DXSDMN+DXSDN1;
end
for N=1:3
      DXSDN2=DXSDN2+(N-1)*(N-2)*D(5,N)*(X4^(N-3));
end
DXSDN2=(Y^32)*DXSDN2;
[DFPR3]=dfpr3(X4);
DFPR4=DFPR3+DXSDMN-DXSDN2;
return
endfunction
```

I.2.9 Function dfps (αρχείο dfps.sci)

```
function[DFPS]=dfps(TS)
//
//    COMPUTE 1st DERIVATIVE OF LIGHT WATER SATURATION PRESSURE BY
//    TEMPERATURE
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    REDUCE SATURATION TEMPERATURE
//
      TR=TS/TC;
//
//    CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
//
      PPC=PC/10.;
      SUMKN=0.;
      DSUMKN=0.;
for N=1:5
      SUMKN=SUMKN+K(N)*((1-TR)^N);
      DSUMKN=DSUMKN+N*K(N)*((1-TR)^(N-1));
end
      PRK1=(1/TR)*SUMKN;
      PRK2=1+K(6)*(1-TR)+K(7)*((1-TR)^2);
      PRK3=K(8)*((1-TR)^2)+K(9);
      PRK4=PRK1/PRK2-(1-TR)/PRK3;
      FPS=exp(PRK4)*PPC;
      DPRK1=(-1/TR)*PRK1/PRK2;
      DPRK2=DPRK1+(-DSUMKN*PRK2+SUMKN*(K(6)+2*K(7)*...
            (1-TR)))/TR/(PRK2^2);
      DPS=DPRK2+(-PRK3+(1-TR)*(2*K(8)*(1-TR)))/(PRK3^2);
      DFPS=FPS*DPS/TC;
      return
endfunction
```

I.2.10 Function energy (αρχείο energy.sci)

```
function[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD)
//
//    COMPUTE SPECIFIC INTERNAL ENERGY OF LIGHT WATER IN kJ/kg GIVEN
//    TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
U=PSI0-TT*PSI0T+R*1000*DD*QTD;
PSI=PSI0+R*TT*(log(DD)+DD*Q);
return
endfunction
```

I.2.11 Function enthalpy (αρχείο enthalpy.sci)

```
function[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD)
//
//    COMPUTE ENTHALPY OF LIGHT WATER IN kJ/kg GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
H=R*TT*(1+DD*Q+DD*(1000/TT)*QTD+DD*DD*QDT)+PSI0-TT*PSI0T;
return
endfunction
```

I.2.12 Function entropy (αρχείο entropy.sci)

```
function[S]=entropy(TT,DD,PSI0T,Q,QTD)
//
//    COMPUTE ENTROPY OF LIGHT WATER IN kJ/(kgK) GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
S=-R*(log(DD)+DD*Q-DD*(1000/TT)*QTD)-PSI0T;
return
endfunction
```

I.2.13 Script example.sci

```
"program example";
funcprot(0);
getd('F:\scilab_light_water\functions');
//
//MASTER TEST PROGRAM FOR LIGHT WATER PROPERTY PACKAGE
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
write(%io(2),'National Technical University of Athens')
write(%io(2),'Department of Mechanical Engineering')
write(%io(2),'Nuclear Engineering Section')
write(%io(2),' ')
write(%io(2),'LIGHT W.A.S.P. (R)')
write(%io(2),'LIGHT WATER AND STEAM PROPERTIES PACKAGE')
write(%io(2),'Thermodynamic and Transport Properties')
write(%io(2),'of the Light Water Substance;an')
write(%io(2),'interactive approach')
write(%io(2),'SCILAB Version 1-SEPTEMBER 2010')
write(%io(2),'Author: STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>')
write(%io(2),'This version of code accepts')
write(%io(2),'as input only pressure and tempreture')
write(%io(2),'(range: 0.006-1000bar,0.01-1000 C)')
write(%io(2),'To obtain a saturation state, equal either')
write(%io(2),'pressure or temperature to zero(for the')
write(%io(2),'desired temperature or pressure,respectively')
write(%io(2),' ')
SUPER=%f;
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
//
PP=0.; PR=0.; TT=0.; TR=0.; DD=0.; JJ=0.;
disp(' ')
disp('Please enter desired pressure in bars:')
P=input('P:');
disp('Please enter desired temperature in C:')
T=input('T:');
if T~=0
      T=T+273.15;
end
JS=1;
JP=63;
[P,T,TS,SVF,SVG,JR,IS]=light_wasp(JS,JP,P,T);
if JR==0
      return
end
printf('Pressure in bars:%8.3f\n',P);
printf('Temperature in C:%8.3f\n',T-273.15);
disp(' ')
if P>PC&T>TC
      SUPER=%t;
      disp('SUPERCRITICAL STATE ')
end
if P<=PC&SVF~=0&IS~=2
      disp('SUBCOOLED LIQUID REGION ')
end
if P<=PC&SVG~=0&IS~=2
      disp('SUPERHEATED STEAM REGION ')
end
```

```
if IS==0
      disp('There is no saturation state for this pressure and temperature')
      disp(' ')
      disp(' ')
end
if IS==1
      disp('Exact saturation temperature for this pressure')
      printf('%7.3f\n',TS-273.15);
      disp(' ')
end
if IS==2
      disp('This is a saturation state or almost a saturation state')
      disp(' ')
      disp(' ')
      disp(' ')
end
if SUPER==%t;
      disp('DENSE PHASE ')
if SVF>0
disp(' ')
      printf('Specific Volume in m3/kg:%8.7f\n',SVF);
      printf('Internal Energy in KJ/Kg:%6.1f\n',UF);
      printf('Specific Enthalpy in KJ/Kg:%6.1f\n',HF);
      printf('Specific Entropy in KJ/KgK:%6.4f\n',SF);
      printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
      printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVF);
      printf('Viscosity in kg/(ms):%8.7f\n',MUF);
      printf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      printf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
      disp(' ')
      return
end
if SVG>0
      disp(' ')
      printf('Specific Volume in m3/kg:%10.6f\n',SVG);
      printf('Internal Energy in kJ/Kg:%6.1f\n',UG);
      printf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      printf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      printf('Viscosity in kg/(ms):%8.7f\n',MUG);
      printf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      printf('Prandtl number:%6.3f\n',PRANDTLG);
      disp(' ')
      disp(' ')
      return
end
end
      disp('LIQUID ')
      disp(' ')
      printf('Specific Volume in m3/kg:%8.7f\n',SVF);
      printf('Internal Energy in kJ/kg:%6.1f\n',UF);
      printf('Specific Enthalpy in kJ/kg:%6.1f\n',HF);
      printf('Specific Entropy in kJ/kgK:%6.4f\n',SF);
      printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
      printf('Specific Heat atconstant volume in kJ/(kgK):%6.4f\n',CVF);
      printf('Viscosity in kg/(ms):%8.7f\n',MUF);
      printf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      printf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
```

```
      disp('VAPOR')
      disp(' ')
      printf('Specific Volume in m3/kg:%10.6f\n',SVG);
      printf('Internal Energy in kJ/kg:%6.1f\n',UG);
      printf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      printf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      printf('Viscosity in kg/(ms):%8.7f\n',MUG);
      printf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      printf('Prandtl number:%6.3f\n',PRANDTLG);
      write(%io(2),' ')
      write(%io(2),' ')
      write(%io(2),' ')
if IS==2
      printf('Surface tension in N/m:%7.6f\n',SIGMAFG);
else
      disp(' ')
      return
end
```

I.2.14 Function fp (αρχείο fp.sci)

```
function[FP]=fp(ZZ)
//
//    COMPUTE PRESSURE OF LIGHT WATER IN MPa GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
qmust(ZZ);
//
//    CONVERT PRESSURE IN BARS TO PRESSURE IN MPa
//
PPP=PP/10.;
FP=ZZ*R*TT*(1+ZZ*Q+ZZ*ZZ*QDT)-PPP;
return
endfunction
```

I.2.15 Function fpr3 (αρχείο fpr3.sci)

```
function[FPR3]=fpr3(X3)
//
//    COMPUTE REDUCED PRESSURE OF LIGHT WATER VAPOR IN THERMODYNAMIC
//    SUB-REGIONS 3 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
//    INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
XSC0N=0.;
XSC1N=0.;
XSC2N=0.;
XSC3N=0.;
XSC4N=0.;
XSC6N=0.;
for N=0:9
      if N>=2
            XSC0N=XSC0N+(1-N)*C(N)/(X3^N);
      end
      if N>=2 & N<=6
            XSC1N=XSC1N+(1-N)*C(10+N)/(X3^N);
      end
      if N>=2 & N<=7
            XSC2N=XSC2N+(1-N)*C(20+N)/(X3^N);
      end
      if N>=2
            XSC3N=XSC3N+(1-N)*C(30+N)/(X3^N);
      end
      if N<=4
            XSC6N=XSC6N+C(60+N)/(TR^(N+2));
      end
end
XSC0N=XSC0N+(1-10)*C010/(X3^10);
XSC0N=XSC0N+(1-11)*C011/(X3^11);
XSC0N=XSC0N+C(1)+C012/X3;
XSC1N=XSC1N+C(11)+C(17)/X3;
XSC2N=XSC2N+C(21)+C(28)/X3;
XSC3N=XSC3N+C(31)+C310/X3;
XSC4N=5*C(41)*(TR-1)/((X3^6)*(TR^23));
XSC6N=6*(X3^5)*XSC6N;
FPR3=-XSC0N-XSC1N*(TR-1)-XSC2N*((TR-1)^2)-XSC3N*((TR-1)^3) ...
      +XSC4N-XSC6N-PR ;
return
endfunction
```

I.2.16 Function fpr4 (αρχείο fpr4.sci)

```
function[FPR4]=fpr4(X4)
//
//    COMPUTE REDUCED PRESSURE OF LIQUID LIGHT WATER IN THERMODYNAMIC
//    SUB-REGIONS 4 AND 5 CORRESPONDING TO REDUCED TEMPERATURE  AND
//    INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
Y=(1-TR)/(1-TR1);
XSDN1=0.D+0;
XSDN2=0.D+0;
XSDMN=0.D+0;
for M=3:4
      for N=1:5
            XSDN1=XSDN1+(N-1)*D(M,N)*(Y^M)/(X4^N);
      end
            XSDMN=XSDMN+XSDN1;
end
for N=1:3
      XSDN2=XSDN2+(N-1)*D(5,N)*(X4^(N-2));
end
XSDN2=(Y^32)*XSDN2;
[FPR3]=fpr3(X4);
FPR4=FPR3+XSDMN-XSDN2;
return
endfunction
```

I.2.17 Function fps (αρχείο fps.sci)

```
function[FPS]=fps(TS)
//
//    COMPUTE SATURATION VAPOR PRESSURE OF LIGHT WATER IN MPa AS
//    FUNCTION OF REDUCED SATURATION TEMPERATURE
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    REDUCE SATURATION TEMPERATURE
//
      TR=TS/TC;
//
//    CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
//
      PPC=PC/10.;
      PPP=PP/10.;
      SUMKN=0.;
for N=1:5
      SUMKN=SUMKN+K(N)*((1-TR)^N);
end
      PRK=(1/TR)*SUMKN;
      PRK=PRK/(1+K(6)*(1-TR)+K(7)*((1-TR)^2));
      PRK=PRK-(1-TR)/(K(8)*((1-TR)^2)+K(9));
      FPS=exp(PRK)*PPC-PPP;
      return
endfunction
```

I.2.18 Function fts (αρχείο fts.sci)

```
function[FTS]=fts
//
//    COMPUTE FIRST APPROXIMATION OF LIGHT WATER SATURATION
//    TEMPERATURE IN K UNITS AS FUNCTION OF PRESSURE IN BARS
//    ACCORDING TO AUTHORS FIT.THIS APPROXIMATION IS TO BE USED
//    AS INITIAL GUESS FOR NUMERICAL CALCULATION OF ACTUAL
//    SATURATION TEMPERATURE OF LIGHT WATER
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
      A=219.88851D+0;
      B=146.70687D+0;
      C=0.19753D+0;
      FTS=A+B*(PP^C);
      return
endfunction
```

I.2.19 Function ibm_jtc (αρχείο ibm_jtc.sci)

```
function[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE ISOTHERMAL BULK MODULUS IN 1/MPa AND JOULE-THOMMSON
//    COEFFICIENT IN K/MPa
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
JTC=(1/(DD*CP))*((TT/DD)*(PTD/PDT)-1);
IBM=1./(DD*PDT);
return
endfunction
```

I.2.20 Script initial.sci

```
'INITIAL';
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
//    REGARDLESS OF PHASE

 D=0.; B2=0.; B3=0.; B4=0.; DB2=0.; DB3=0.;
 DB4=0.; U=0.; PSI=0.; H=0.; S=0.; CP=0.;
 CV=0.; IBM=0.;JTC=0.; GAMMA=0.; A=0.;
 MU=0.; K=0.; PRANDTL=0.; SIGMA=0.;
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY
//    WATER SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
//

DF=0.; B2F=0.; B3F=0.; B4F=0.; DB2F=0.; DB3F=0.; DB4F=0.;
 UF=0.; PSIF=0.; HF=0.; SF=0.; CPF=0.; CVF=0.; IBMF=0.; JTCF=0.;
GAMMAF=0.; AF=0.; MUF=0.; KF=0.; PRANDTLF=0.;

//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
//    VAPOR SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
//

DG=0.; B2G=0.; B3G=0.; B4G=0.; DB2G=0.; DB3G=0.; DB4G=0.;
UG=0.; PSIG=0.; HG=0.; SG=0.; CPG=0.; CVG=0.; IBMG=0.; JTCG=0.;
GAMMAG=0.; AG=0.; MUG=0.; KG=0.; PRANDTLG=0.;

//
//    VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN
//    SATURATION STATES,SUFFIX FG STANDS FOR SATURATION
//

HFG=0.; SIGMAFG=0.; LACFG=0.;

//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT  QUANDITIES FOR THE
//    MAIN DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE
//    SERIES EXPANSION OF Q
//

Q=0.; QDT=0.; Q2D2T=0.; QTD=0.; Q2T2D=0.;
Q2DT=0.; PSI0=0.; PSI0T=0.; PSI02T2=0.;
```

I.2.21 Function light_wasp (αρχείο light_wasp.sci)

```
function [P,T,TS,SVF,SVG,JR,IS] =light_wasp(JS,JP,P,T)
IER=0;
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    INITIALIZE VALUES OF DENSITIES AND SPECIFIC VOLUMES
//
DF=0.; DG=0.; SVF=0.; SVG=0.;
//
//    INITIALIZE VALUES OF OTHER PROPERTIES
//
exec('F:\scilab_light_water\initial.sci');
//
//    LIGHT WATER AND STEAM PROPERTY PACKAGE DEVELOPED BY
//    STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>
//
//    COMPUTE THERMODYNAMIC AND TRANSPORT PROPERTIES OF LIGHT WATER
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN DEGREES CENTIGRADE
//    MAINLY ACCORDING TO EQUATION OF STATE DEVELOPED BY HILL P.G.,
//    MACMILLAN R.D.C. AND LEE V.,AND EQUATIONS FOR TRANSPORT
//    PROPERTIES DEVELOPED BY MATSUNAGA N. AND NAGASHIMA A.
//    PARTS OF CODE STRUCTURE ARE BASED TO SIMILAR PROPERTY PACKAGE
//    FOR LIGHT WATER DEVELOPED BY HENDRICKS R.C.,PELLER I.C. AND
//    BARON A.K.PARTS OF EQUATION OF STATE FOR LIGHT WATER DEVELOPED
//    BY SCHMIDT E. AND GRIGULL U. ARE USED TO PROVIDE GOOD INITIAL
//    GUESSES FOR NUMERICAL CALCULATION OF HEAVY WATER DENSITY.
//    VALUE OF JS(INPUT) MUST BE 1.CALCULATED THERMODYNAMIC AND
//    TRANSPORT PROPERTIES ARE SPECIFIED BY JP(INPUT).JR(OUTPUT)
//    REPRESENTS THERMODYNAMIC SUB-REGION OF P-T DIAGRAMM IN WHICH
//    INPUT LIES.IS(OUTPUT) INDICATES RELATIVE POSITION OF INPUT TO
//    SATURATION CURVE.COMPUTE SATURATION PROPERTIES IF EITHER PRESSURE
//    OR TEMPERATURE IS EQUAL TO ZERO (FOR RESPECTIVE TEMPERATURE OR
//    PRESSURE)
//
//    CHECK FOR CORRECT CALL OF LIGHT_WASP ROUTINE
//
if JS~=1
      disp('Internal Error: Illegal JS parameter')
      return
end
if JP>63 | JP<0
      disp('Internal Error: Illegal JS parameter')
      return
end
PP=P;
TT=T;
TS=0.;
//
//    CHECK FOR OUT OF RANGE
//
[TS,JR,IS]=checkpt();
if JR==0
      return
end
P=PP;
T=TT;
JJ=JR;
//
```

```
//    DENSITY OF LIQUID HEAVY WATER
//
if JJ==1 | JJ==6 | JJ==4 | JJ==5
      [DF,IER]=densf();
      if DF~=0.
//
//    CONVERT DENSITY TO SPECIFIC VOLUME IN m3/Kg
//
      SVF=1/(1000*DF);
      DD=DF;
      end
//
//    CHECK FOR CRITICAL POINT VICINITY
//
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have may not be accurate!')
      return
      end
//
//    OTHER PROPERTIES OF LIQUID LIGHT WATER
//
if JP~=0
[B2F,B3F,B4F,DB2F,DB3F,DB4F,UF,PSIF,HF,SF,CPF,CVF,IBMF,JTCF,GAMMAF,...
AF,MUF,KF,PRANDTLF,SIGMAFG]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
end
end
//
//    DENSITY OF LIGHT WATER VAPOR
//
if JJ==2 | JJ==6 | JJ==3 | JJ==5
      [DG,IER]=densg();
      if DG~=0.
//
//    CONVERT DENSITY TO SPECIFIC VOLUME IN m3/kg
//
      SVG=1/(1000*DG);
      DD=DG;
      end
//
//    CHECK FOR CRITICAL POINT VICINITY
//
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have no validity!')
      return
      end
//
//    OTHER PROPERTIES OF HEAVY WATER VAPOR
//
if JP~=0
[B2G,B3G,B4G,DB2G,DB3G,DB4G,UG,PSIG,HG,SG,CPG,CVG,IBMG,JTCG,GAMMAG,...
AG,MUG,KG,PRANDTLG,SIGMAFG]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
end
end
if JJ==5 | JJ==6
//
//    COMPUTE LATENT HEAT OF HEAVY WATER IN kJ/kg
//
      HFG=HG-HF;
//
```

```
//    COMPUTE LAPLACE CONSTANT OF HEAVY WATER IN m
//
      [DF,IER]=densf();
      [DG,IER]=densg();
      if DF~=0. & DG~=0. & abs(DF-DG)>=1.D-3
            G=9.80665D+0;
            LACFG=sqrt(SIGMAFG/(G*1000*(DF-DG)));
      end
end
disp(' ')
disp(' ')
disp(' ')
return
endfunction
```

I.2.22 Function newton (αρχείο newton.sci)

```
function[X,IER]=newton(f,df,EPS,NDEC,X,ITMAX)
//
//    FIND REAL ZERO X OF REAL FUNCTION F GIVEN GOOD INITIAL GUESS
//    X AND FUNCTION DF (1st PARTIAL DERIVATIVE OF F BY X).ROOT X
//    IS ACCEPTED IF ABSOLUTE VALUE OF F(X).LE.EPS(INPUT).ROOT X
//    IS ACCEPTED IF TWO SUCCESSIVE APPROXIMATIONS AGREE TO FIRST
//    NDEC(INPUT) DECIMAL DIGITS.X(INPUT) IS AN INITIAL
//    VALUE.X(OUTPUT) CONTAINS COMPUTED ROOT.ITMAX(INPUT) IS...
//    MAXIMUM ALLOWBLE NUMBER OF NEWTON_RAPHSON ITERATIONS USED...
//    ON ROOT.IER(OUTPUT) WARNING ERROR PARAMETER=N,N=1 ROOT X ...
//    WAS BYPASSED BECAUSE DF BECOMES TOO SMALL.X IS SET TO ...
//    111111.THIS ERROR CONDITION MAY CAUSE AN
//    OVERFLOW,N=2 ROOT X WAS BYPASSED  BECAUSE ITMAX WAS EXCEEDED.
//    X IS SET TO 222222,N=3 SEVERAL OF THE ABOVE ERROR CONDITIONS
//    OCCURED.X IS SET TO EITHER 111111. OR 222222. AS ABOVE.DOUBLE
//    PRECISION ROUTINE.
//
TOL=10.^(-NDEC);
IER=0;
if ITMAX>100
      ITMAX=100;
end
for K=1:ITMAX
      dd=feval(X,df);
      if abs(dd)<EPS
            X=111111.D+0;
            IER=1;
            return
      end
      ff=feval(X,f);
      dd=feval(X,df);
      XK=X-ff/dd;
      if K==41
            TOL=TOL*10;
      end
      if K==61
            TOL=TOL*10;
      end
      if K==81
            TOL=TOL*10 ;
      end
      if abs(XK-X)<TOL
            X=XK;
            return
      else
            X=XK;
            ff=feval(X,f);
            if abs(ff)<EPS
            return
            end
      end
end
      X=222222.D+0;
      if IER==0
            IER=2;
      else
            IER=3;
      end
```

```
      return
endfunction
```

I.2.23 Function prl (αρχείο prl.sci)

```
function[PRL]=prl
//
//    COMPUTE REDUCED PRESSURE OF LIGHT WATER CORRESPONDING TO
//    REDUCED TEMPERATURE OF HEAVY WATER CALCULATED FROM GIVEN
//    TEMPERATURE IN K UNITS ALONG THE BOUNDARY BETWEEN THERMODYNAMIC
//    SUB-REGIONS 2 AND 3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
L=7.160997524D+0;
PRL=((TR2-TR)*PR1+(TR-TR1)*PR2-L*(TR2-TR)*(TR-TR1))/(TR2-TR1);
return
endfunction
```

I.2.24 Function prs (αρχείο prs.sci)

```
function[PRS]=prs
//
//    COMPUTE SATURATION VAPOR PRESSURE OF LIGHT WATER IN MPa AS
//    FUNCTION OF REDUCED SATURATION TEMPERATURE
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
//
      PPC=PC/10.;
      PPP=PP/10.;
      SUMKN=0.D+0;
for N=1:5
      SUMKN=SUMKN+K(N)*((1-TR)^N);
end
      PRK=(1/TR)*SUMKN;
      PRK=PRK/(1+K(6)*(1-TR)+K(7)*((1-TR)^2));
      PRK=PRK-(1-TR)/(K(8)*((1-TR)^2)+K(9));
      PRS=exp(PRK);
      return
endfunction
```

I.2.25 Function qmust (αρχείο qmust.sci)

```
function[]=qmust(ZZ)
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    PRIMARY CONSTANTS OF DOUBLE SERIES EXPANSION OF Q
//
//    COEFFICIENTS Aij
//
A=[29.492937D+0,-5.1985860D+0,6.8335354D+0,-0.1564104D+0,...
-6.3972405D+0,-3.9661401D+0,-0.69048554D+0;...
-132.13917D+0,7.7779182D+0,-26.149751D+0,...
-0.72546108D+0,26.409282D+0,15.453061D+0,2.7407416D+0;...
274.64632D+0,-33.301902D+0,65.326396D+0,-9.2734289D+0, ...
-47.740374D+0,-29.142470D+0,-5.1028070D+0;...
-360.93828D+0,-16.254622D+0,-26.181978D+0,...
4.3125840D+0,56.323130D+0,29.568796D+0,3.9636085D+0;...
342.18431D+0,-177.31074D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0;...
-244.50042D+0,127.48742D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0;...
155.18535D+0,137.46153D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0;...
5.9728487D+0,155.97386D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0,0.0D+0;...
-410.30848D+0,337.31180D+0,-137.46618D+0,...
6.7874983D+0,136.87317D+0,79.847970D+0,13.041253D+0;...
-416.05860D+0,-209.88866,-733.96848D+0,...
10.401717D+0,645.81880D+0,399.17570D+0,71.531353D+0];
//
//    COEFFICIENTS Ci
//
      C=[1855.3865D+0,3278.642D+0,-379.03D+0,46.174D+0,-1021.17];
//
//    OTHER CONSTANTS
//
      TAJ1=1.544912D+0;
      TAJ2=2.5D+0;
      E=4.8D+0;
      RAJ1=0.634D+0;
      RAJ2=1.0D+0;
//
//    TEMPERATURE PARAMETER
//
      TAU=1000/TT;
      TAUC=TAJ1;
      EX=exp(-E*ZZ);
      SUMI1=0.;
      SUMI3=0.;
      SUMI5=0.;
for I=1:8
      SUMI1=SUMI1+A(I,1)*(ZZ-RAJ1)^(I-1);
      if I>=2
            SUMI3=SUMI3+(I-1)*A(I,1)*(ZZ-RAJ1)^(I-2);
      end
      if I>=3
            SUMI5=SUMI5+A(I,1)*(I-1)*(I-2)*(ZZ-RAJ1)^(I-3);
      end
end
      SUMJ1=0.;
      SUMJ2=0.;
      SUMJ3=0.;
      SUMJ4=0.;
```

```
      SUMJ5=0.;
      SUMJ6=0.;
for J=1:7
      SUMI2=0.;
      SUMI4=0.;
      SUMI6=0.;
for I=1:8
      SUMI2=SUMI2+A(I,J)*(ZZ-RAJ2)^(I-1);
      if I>=2
            SUMI4=SUMI4+(I-1)*A(I,J)*(ZZ-RAJ2)^(I-2);
      end
      if I>=3
            SUMI6=SUMI6+A(I,J)*(I-1)*(I-2)*(ZZ-RAJ2)^(I-3);
      end
end
      SUMI2=SUMI2+EX*(A(9,J)+A(10,J)*ZZ);
      SUMI4=SUMI4+EX*(-E*(A(9,J)+A(10,J)*ZZ)+A(10,J));
      SUMI6=SUMI6+EX*(-E)*(-E*A(9,J)+A(10,J)*(2-E*ZZ));
      if J>=2
            SUMJ1=SUMJ1+((TAU-TAJ2)^(J-2))*SUMI2;
      end
      if J>=3
            SUMJ2=SUMJ2+(J-2)*((TAU-TAJ2)^(J-3))*SUMI2;
      end
      if J>=2
            SUMJ3=SUMJ3+((TAU-TAJ2)^(J-2))*SUMI4;
      end
      if J>=3
            SUMJ4=SUMJ4+(J-2)*((TAU-TAJ2)^(J-3))*SUMI4;
      end
      if J>=2
            SUMJ5=SUMJ5+((TAU-TAJ2)^(J-2))*SUMI6;
      end
      if J>=4
            SUMJ6=SUMJ6+(J-2)*(J-3)*((TAU-TAJ2)^(J-4))*SUMI2;
      end
end
      SUMI1=SUMI1+EX*(A(9,1)+A(10,1)*ZZ);
      SUMI3=SUMI3+EX*(-E*(A(9,1)+A(10,1)*ZZ)+A(10,1));
      SUMI5=SUMI5+EX*(-E)*(-E*A(9,1)+A(10,1)*(2-E*ZZ));
//
//    DATA-FITTING FUNCTION,Q
//
      Q=SUMI1+(TAU-TAUC)*SUMJ1;
//
//    1st PARTIAL DERIVATIVE OF Q BY TEMPERATURE
//
      QTD=SUMJ1+(TAU-TAUC)*SUMJ2;
//
//    1st PARTIAL DERIVATIVE OF Q BY DENSITY
//
      QDT=SUMI3+(TAU-TAUC)*SUMJ3;
//
//    2nd PARTIAL DERIVATIVE OF Q BY DENSITY AND TEMPERATURE
//
      Q2DT=SUMJ3+(TAU-TAUC)*SUMJ4;
//
//    2nd PARTIAL DERIVATIVE OF Q BY DENSITY
//
      Q2D2T=SUMI5+(TAU-TAUC)*SUMJ5;
//
```

```
//    2nd PARTIAL DERIVATIVE OF Q BY TEMPERATURE
//
      Q2T2D=2*SUMJ2+(TAU-TAUC)*SUMJ6;
      SUMI7=0.;
      SUMI8=0.;
      SUMI9=0.;
for I=1:3
      SUMI7=SUMI7+C(I)*((TT/1000.)^(I-1));
      SUMI8=SUMI8+C(I)*(I-1)*((TT/1000.)^(I-2));
      SUMI9=SUMI9+C(I)*(I-1)*(I-2)*((TT/1000.)^(I-3));
end
//
//    REFERENCE FUNCTION,PSI0
//
      PSI0=SUMI7+C(4)*log(TT)+C(5)*TT*log(TT)/1000;
//
//    1st PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
//
      PSI0T=SUMI8/1000+C(4)/TT+C(5)*log(TT)/1000+C(5)/1000;
//
//    2nd PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
//
      PSI02T2=SUMI9*(1.D-6)-C(4)/(TT*TT)+C(5)/(1000*TT);
      return
endfunction
```

I.2.26 Function shp (αρχείο shp.sci)

```
function[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE SPECIFIC HEAT OF LIGHT WATER AT CONSTANT PRESSURE IN
//    kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
      PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
return
endfunction
```

I.2.27 Function shr_sove (αρχείο shr_sove.sci)

```
function[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE ISENTROPIC EXPANSION COEFFICIENT(SPECIFIC HEAT RATIO)
//    AND SONIC VELOCITY IN m/s OF HEAVY WATER
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)*...
      Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
//
//    COMPUTE CP
//
CP=HTD-HDT*PTD/PDT;
//
//    COMPUTE CV
//
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
GAMMA=CP/CV;
A=(10^(3/2.))*(1./sqrt(1/PDT-(TT/(CP*(DD^2)))*(PTD^2)/...
      (PDT^2)));
return
endfunction
```

I.2.28 Function shv (αρχείο shv.sci)

```
function[CV]=shv(TT,DD,PSI02T2,Q2T2D)
//
//    COMPUTE SPECIFIC HEAT OF LIGHT WATER AT CONSTANT VOLUME IN
//    kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
return
endfunction
```

I.2.29 Function svlwl (αρχείο svlwl.sci)

```
function[DL,IER]=svlwl
//
//    COMPUTE DENSITY OF LIQUID LIGHT WATER IN g/cm3 CORRESPONDING TO
//    REDUCED PRESSURE AND TEMPERATURE OF LIGHT WATER CALCULATED FROM
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
//
IER=0;
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS IN THERMODYNAMIC
//    SUB-REGION 1
//
AM=[8.438375405D-1 5.362162162D-4 1.72D+0 7.342278489D-2 ...
      4.97585887D-2 6.5371543D-1 1.15D-6 1.5108D-5 ...
      1.4188D-1 7.002753165D+0 2.995284926D-4 2.04D-1];
A=[0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0  0.D+0 ...
      7.982692717D+0 -2.616571843D-2 1.52241179D-3 2.284279054D-2 ...
      2.421647003D+2 1.269716088D-10 2.074838328D-7 2.17402035D-8 ...
      1.105710498D-9 1.293441934D+1 1.308119072D-5 6.047626338D-14];
      EPS=1.D-5; NDEC=5; ITMAX=1000;
//
//    ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
if JJ==6 | JJ==5
      [PRS]=prs();
      PR=PRS;
end
Y=1-(AM(1)*TR*TR)-(AM(2)/(TR^6));
Z=AM(3)*Y*Y-2*AM(4)*TR+2*AM(5)*PR;
if Z<=0
      Z=Y;
else
      Z=Y+sqrt(Z);
end
      X1=A(11)*AM(5)/(Z^(5./17.))...
      +(A(12)+A(13)*TR+A(14)*TR*TR+A(15)*((AM(6)-TR)^10)...
      +A(16)/(AM(7)+(TR^19)))...
      -(A(17)+2*A(18)*PR+3*A(19)*PR*PR)/(AM(8)+(TR^11))...
      -A(20)*(TR^18)*(AM(9)+TR*TR)...
      *(-3/((AM(10)+PR)^4)+AM(11))...
      +3*A(21)*(AM(12)-TR)*PR*PR+4*(A(22)/(TR^20))*(PR^3);
if JJ==4
      X4=X1;
      [X4,IER]=newton(fpr4,dfpr4,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
if JJ==5
      [DV,IER]=svlwv();
      X4=1./(DV*VC);
      X4=X1;
      [X4,IER]=newton(fpr4,dfpr4,EPS,NDEC,X4,ITMAX);
      if X4<=0
      X4=10*R*TT/(PP*VC);
      [X4,IER]=newton(fpr4,dfpr4,EPS,NDEC,X4,ITMAX);
      end
      X1=X4;
end
```

```
PR=PP/PC;
VL=X1*VC;
DL=1/VL;
return
endfunction
```

I.2.30 Function svlwv (αρχείο svlwv.sci)

```
function[DV,IER]=svlwv
//
//    COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 CORRESPONDING TO
//    REDUCED PRESSURE AND TEMPERATURE OF LIGHT WATER CALCULATED FROM
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
IER=0;
exec('F:\scilab_light_water\constants.sci');
exec("F:\scilab_light_water\transient.sci');
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS
//
//    THERMODYNAMIC SUB-REGION 2
//
      B0=7.633333333D-1; B90=1.936587558D+2;
      B=[6.670375918D-2,1.388983801D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
      8.390104328D-2,2.614670893D-2,-3.373439453D-2,0.D+0,0.D+0,...
      0.D+0;4.520918904D-1,1.069036614D-1,0.D+0,0.D+0,0.D+0,...
      0.D+0;-5.975336707D-1,-8.847535804D-2,0.D+0,0.D+0,0.D+0,...
      0.D+0;5.958051609D-1,-5.159303373D-1,2.075021122D-1,...
      0.D+0,0.D+0,0.D+0;1.190610271D-1,-9.867174132D-2,...
      0.D+0,0.D+0,0.D+0,0.D+0;1.683998803D-1,-5.809438001D-2,...
      0.D+0,0.D+0,0.D+0,0.D+0;6.552390126D-3,5.710218649D-4,...
      0.D+0,0.D+0,0.D+0,0.D+0;-1.388522425D+3,4.126607219D+3, ...
      -6.508211677D+3,5.745984054D+3,-2.693088365D+3,5.235718623D+2];

      BM(6,1)=4.006073948D-1; BM(7,1)=8.636081627D-2;
      BM(8,1)=-8.532322921D-1; BM(8,2)=3.460208861D-1;
//
//    NUMBERS OF TERMS n(m) AND l(m),AND EXPONENTS z(m,n) AND x(m,n)
//
      NM=[2,3,2,2,3,2,2,2];
      ZM=[13,3,0;
            18,2,1;
            18,10,0;
            25,14,0;
            32,28,24;
            12,11,0;
            24,18,0;
            24,14,0];
      LM=[0,0,0,0,0,1,1,2];
      XM=[0,0;0,0;0,0;0,0;0,0;14,0;19,0;54,27];
      EPS=1.D-5;
      NDEC=5;
      ITMAX=100;
//
//    ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
if JJ==6 | JJ==5
      [PRS]=prs();
      PR=PRS;
end
I1=10*R*TC/(PC*VC);
X=exp(B0*(1-TR));
VSM1=0.D+0;
VSM2=0.D+0;
VSN=0.D+0;
for M=1:8
      VSN1=0.D+0;
      VSN2=0.D+0;
```

```
      VSL=0.D+0;
      if M<=5
      for N=1:NM(M)
            VSN1=VSN1+B(M,N)*(X^ZM(M,N));
      end
            VSM1=VSM1+M*(PR^(M-1))*VSN1;
      else
      for N=1:NM(M)
            VSN2=VSN2+B(M,N)*(X^ZM(M,N));
      end
      for L=1:LM(M)
            VSL=VSL+BM(M,L)*(X^XM(M,L));
      end
            VSM2=VSM2+((M-2)*(PR^(1-M))*VSN2)/(((PR^(2-M))+VSL)^2);
      end
      end
[PRL]=prl();
VSN=VSN+(11.*(PR/PRL)^10)*B90;
for N=1:6
      [PRL]=prl();
      VSN=VSN+(11.*(PR/PRL)^10)*B(9,N)*(X^N);
end
X2=I1*(TR/PR)-VSM1-VSM2+VSN;
if JJ==3 | JJ==5
      if X2<=0
            X3=I1*(TR/PR);
      else
            X3=X2;
      end
      [X3,IER]=newton(fpr3,dfpr3,EPS,NDEC,X3,ITMAX);
      X2=X3;
end
PR=PP/PC;
VV=VC*X2;
DV=1/VV;
return
endfunction
```

I.2.31 Function tension (αρχείο tension.sci)

```
function[SIGMA]=tension(TT,JJ)
//
//    COMPUTE SURFACE TENSION OF LIGHT WATER IN N/m GIVEN
//    TEMPERATURE IN K UNITS ACCORDING TO EQUATION DEVELOPED BY
//    STRAUB J.,ROSNER N. AND GRIGULL U.
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
      SIGMA0=235.8D-3; M=1.256D+0; B=-0.625D+0;
      TCSTR=647.15D+0;
      SIGMA=0.;
      if JJ~=6 & JJ~=5
            return
      end
      TAUT=1-TT/TCSTR;
      SIGMA=SIGMA0*(TAUT^M)*(1+B*TAUT);
      return
endfunction
```

I.2.32 Function total (αρχείο total.sci)

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
A,MU,KK,PRANDTL,SIGMA]=total(JP,J,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    MASTER SERVICE SUBPROGRAM DIRECTLY UNDER LIGHT_WASP ROUTINE.
//    IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
//    MUST BE CALLED TWICE.
//
//    ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
//    SPECIFICATION,JP
//
exec('F:\scilab_light_water\transient.sci');
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 42 43...
       46 47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44 ...
      45 46 47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44 ...
      45 46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51 ...
      52 53 54 55 56 57 58 59 60 61 62 63];
D=0.; B2=0.; B3=0.; B4=0.; DB2=0.; DB3=0.;
DB4=0.; U=0.; PSI=0.; H=0.; S=0.; CP=0.;
CV=0.; IBM=0.;JTC=0.; GAMMA=0.; A=0.;
MU=0.; K=0.; PRANDTL=0.; SIGMA=0.;
//
//    VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
//    TEMPERATURE AND INTERNAL ENERGY
//
function T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70();
return
endfunction
function[]=T70
for I=1:32
      if JP-JPC1(I)<0
      T100();
      return
      end
      if JP-JPC1(I)==0
      T90();
      return
      end
      end
return
endfunction
//
//    ENTHALPY AND ENTROPY
//
function[]=T90
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
T100();
return
endfunction
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
```

```
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120();
      return
      end
end
return
endfunction
//
//    SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT
//    VOLUME, SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
//    MODULUS AND JOULE-THOMSON COEFFICIENT
//
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130();
return
endfunction
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150();
      return
      end
end
return
endfunction
//
//    VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
//
function[]=T150
[MU]=viscosity(TT,DD);
[KK]=conductive(TT,DD);
if KK~=0.D+0
      exec('F:\scilab_light_water\transient.sci');
      PRANDTL=1000.*MU*CP/KK;
end
T160();
return
endfunction
function[]=T160
for I=1:32
      if JP-JPC4(I)<0
      T190();
      return
      end
      if JP-JPC4(I)==0
      T180();
      return
      end
end
return
```

```
endfunction
//
//    Surface tension
//
function[]=T180
      if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
      end
      T190();
      return
endfunction
function[]=T190
      if JP-32<0
      T200();
      return
      end
      if JP-32==0
      T210();
      return
      end
      if JP-32>0
      T210();
      T200();
      end
      return
endfunction
function[]=T200
      return
endfunction
function[]=T210
      return
endfunction
if pmodulo(JP,2)<0
      T60();
end
if pmodulo(JP,2)==0
      T70();
      T100();
end
if pmodulo(JP,2)>0
      T60();
end
return
endfunction
```

I.2.33 Script transient.sci

```
"TRANSIENT";
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
//
global PP PR TT TR DD JJ

global D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC
global GAMMA A MU KK PRANDTL SIGMA
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY
//    WATER SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
//
global  DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF JTCF
global  GAMMAF AF MUF KF PRANDTLF
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
//    VAPOR SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
//
global DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG JTCG
global GAMMAG AG MUG KG PRANDTLG
//
//    VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN
//    SATURATION STATES, SUFFIX FG STANDS FOR SATURATION
//
global HFG SIGMAFG LACFG
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE
//    MAIN DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE
//    SERIES
//    EXPANSION OF Q
global Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2
```

I.2.34 Function viscosity (αρχείο viscosity.sci)

```
function[MU]=viscosity(TT,DD)
//
//    COMPUTE VISCOSITY OF LIGHT WATER IN kg/(ms) GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//    FROM AN EQUATION SUITABLE FOR INDUSTRIAL USE
//
exec('F:\scilab_light_water\constants.sci');
exec('F:\scilab_light_water\transient.sci');
      HR=[1.00000D+0,0.978197D+0,0.579829D+0,-0.202354D+0];
HHR=[0.5132047D+0,0.2151778D+0,-0.2818107D+0,0.1778064D+0,...
-0.04176610D+0,0.D+0,0.D+0;0.3205656D+0,0.7317883D+0,...
-1.070786D+0,0.4605040D+0,0.D+0,-0.01578386D+0,0.D+0;...
0.D+0,1.241044D+0,-1.263184D+0,0.2340379D+0,0.D+0,0.D+0,0.D+0;...
0.D+0,1.476783D+0,0.D+0,-0.4924179D+0,0.1600435D+0,0.D+0,...
-0.003629481D+0;-0.7782567D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
0.1885447D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0];
      NC=55.071D-6;
      PSEUDODR=DD/PSEUDODC;
      PSEUDOTR=TT/PSEUDOTC;
      NR0=0.D+0;
for I=1:4
      NR0=NR0+HR(I)/(PSEUDOTR^(I-1));
end
      NR0=sqrt(PSEUDOTR)/NR0;
      NR1=0.D+0;
for J=1:6
for K=1:7
      NR1=NR1+HHR(J,K)*(((1./PSEUDOTR)-1)^(J-1))*((PSEUDODR-1)^(K-1));
end
end
      NR1=exp(PSEUDODR*NR1);
      NR2=1.D+0;
      NR=NR0*NR1*NR2;
      MU=NR*NC;
      return
endfunction
```

ΠΑΡΑΡΤΗΜΑ ΙΙ: Κώδικας βαρέος ύδατος

ΙΙ.1 Περιβάλλον προγραμματισμού MATLAB

Τα υποπρογράμματα παρατίθενται κατά την αλφαβητική σειρά της ονομασίας τους. Συμπεριλαμβάνεται το κυρίως πρόγραμμα example.m.

ΙΙ.1.1 Function checkpt (αρχείο checkpt.m)

```
function[TS,JR,IS]=checkpt
IER=0;
TS=0.;
JR=0;
IS=0;
constants;
transient;
EPS=1.D-5; NDEC=5; ITMAX=100;
%
%     CHECK PRESSURE FOR OUT OF RANGE
%
if (PP<PMIN | PP>PMAX) & (PP~=0.)
      disp(' ')
      disp('Desired Pressure out of range')
      disp(' ')
      return
end
%
%     CHECK TEMPERATURE FOR OUT OF RANGE
%
if (TT<TMIN | TT>TMAX) & (TT~=0.)
      disp(' ')
      disp('Desired Temperature out of range')
      disp(' ')
      return
end
%
%     IF PRESSURE AND TEMPERATURE ARE EQUAL TO 0 CHECK FOR CORRECT CALL
%     OF SATURATION PROPERTIES
%
if PP==0. & TT==0.
      disp(' ')
      disp('Both pressure and temperature cannot be 0.!')
      disp(' ')
      return
end
%
%     IF PRESSURE OR TEMPERATURE IS EQUAL TO 0 CHECK FOR OUT OF
%     SATURATION RANGE
%
if (PP>PC & TT==0.) | (TT>TC & PP==0.)
      disp(' ')
      disp('Such a saturation state does not exist!')
      disp(' ')
      return
end
if TT==0.;
      TT=TT+273.15D+0;
end
if PP==0.;
```

```
      [FPS]=fps(TT);
      PP=FPS*10;
      TS=TT;
      IS=1;
else
      if PP<=PC & TT<=TC
            [FTS]=fts;
            TS=FTS;
            [FPS]=fps(TS);
            if abs(FPS)>EPS
                  f='fps';
                  df='dfps';
                  [TS,IER]=newton(f,df,EPS,NDEC,TS,ITMAX);
                  IS=1;
                  if TT==273.15D+0
                        TT=TS;
                  end
            end
      end
end
if abs(TT-TS)<=1.D+0
      IS=2;
end
%
%     REDUCED PRESSURE AND TEMPERATURE
%
PR=PP/PC;
TR=TT/TC;
%
%     DETERMINE THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
[PRS]=prs();
if TR>=TRT & TR<=TR1
      if abs(TT-TS)<=1.D-5
            JR=6;
            return
      else
            if PR>=0. & PR<PRS
                  JR=2;
                  return
            end
            if PR>PRS & PR<=PR2
                  JR=1;
                  return
            end
      end
end
if TR>TR1 & ((TR-1.)<1.D-5)
      if abs(TT-TS)<=1.D-5
            JR=5;
            return
      end
else
      [PRS]=prs();
      [PRL]=prl();
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<PRS
            JR=3;
```

```
            return
      end
      if PR>PRS & PR<PR2
            JR=4;
            return
      end
end
if (abs(TR-1.)>=1.D-5) & TR<TR2
      [PRL]=prl();
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<=PR2
            JR=3;
            return
      end
end
if TR>=TR2 & TR<=TR3
      JR=2;
      return
end
if JR==0
      disp(' ')
      disp('Internal Error: JR indicator value is zero')
      disp(' ')
      return
end
return
end
```

II.1.2 Function conductive (αρχείο conductive.m)

```
function[K]=conductive(TR,DD)
%
%     COMPUTE THERMAL CONDUCTIVITY OF HEAVY WATER IN W/(mK) GIVEN
%     TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%     COMPUTATION AT VICINITY OF CRITICAL POINT IS NOT ACCURATE WITH
%     THIS FIT
%
constants;
A0=1.00000D+0;
A=[37.3223D+0 22.5485D+0 13.0465D+0 0.D+0 -2.60735D+0];
B=[483.65600D+0 -191.039D+0 73.0358D+0 -7.57467D+0];
BE=-2.506D+0;
B0=-167.31D+0;
CT1=0.144847D+0;
CT2=-5.64493D+0;
CR1=-2.80000D+0;
CR2=-0.080738543D+0;
CR3=-17.9430D+0;
PSEUDODR1=0.125698D+0;
C1=0.354296D+5;
C2=0.5D+10;
D1=-741.112D+0;
LAMDA=0.742128D-3;
PSEUDODR=DD/PSEUDODC;
EL0=A0;
for I=1:5
      EL0=EL0+A(I)*(TR^I);
end
DELTAEL=0.D+0;
for J=1:4
      DELTAEL=DELTAEL+B(J)*(PSEUDODR^J);
end
DELTAEL=DELTAEL+B0*(1-exp(BE*PSEUDODR));
F1=exp(CT1*TR+CT2*(TR^2));
F2=exp(CR1*((PSEUDODR-1)^2))+CR2*exp(CR3*((PSEUDODR-PSEUDODR1)^2));
TAUR=TR/(abs(TR-1.1)+1.1);
F3=1+exp(60*(TAUR-1)+20);
F4=1+exp(100*(TAUR-1)+15);
DELTAELC=1+(F2^2)*(C2*(F1^4)/F3+3.5*F2/F4);
DELTAELC=C1*F1*F2*DELTAELC;
DELTAELL=D1*(F1^1.2)*(1-exp(-((PSEUDODR/2.5)^10)));
EL=LAMDA*(EL0+DELTAEL+DELTAELC+DELTAELL);
K=EL;
return
end
```

II.1.3 Script constants.m

```
'CONSTANTS';
%
%     SCRIPT CONTAINING ALL CONSTANTS OF LIGHT_WASP
%     BEING USED AT LEAST TWICE.ALL OTHER ONCE USED CONSTANTS ARE
%     PART OF CORRESPONDING ASSIGNMENTS IN INDIVIDUAL SUBPROGRAMS
%
global PMIN PMAX TMIN TMAX PC TC VC TRT TR1 TR2 TR3 PR1 PR2 R A1 A2 A4 A11
global A20 K C C010 C011 C012 C310 D PSEUDODC
%
%     PRESSURE AND TEMPERATURE RANGE OF EQUATIONS FOR THERMODYNAMIC AND
%     TRANSPORT PROPERTIES OF HEAVY WATER
%
PMIN=0.006601D+0; PMAX=1000.D+0; TMIN=276.95D+0; TMAX=775.D+0;
%
%     CRITICAL PRESSURE TEMPERATURE AND SPECIFIC VOLUME OF HEAVY WATER
%
PC=216.6D+0; TC=643.89D+0; VC=2.793D+0;
%
%     REDUCED TEMPERATURE RANGES OF THERMODYNAMIC SUB REGIONS
%
TRT=4.30120051D-1; TR1=9.626911787D-1;
TR2=1.333462073D+0; TR3=1.657886606D+0;
%
%     MAXIMUM REDUCED PRESSURE
%
PR1=7.454108957D-1; PR2=4.616805171D+0;
%
%     HEAVY WATER CONSTANT
%
R=0.415147D+0;
%
%     CONSTANTS OF HEAVY WATER SATURATION LINE
%
A1=-7.81583D+0; A2=17.6012D+0; A4=-18.1747D+0; A11=-3.92488D+0;
A20=4.19174D+0;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS OF EQUATION OF STATE OF
%     LIGHT WATER BEING USED IN THIS PACKAGE
%
%     SATURATION LINE
%
K=[-7.691234564D+0 -2.608023696D+1 -1.681706546D+2 6.423285504D+1 ...
      -1.189646225D+2 4.167117320D+0 2.097506760D+1 1.D+9 6.D+0];
%
%     THERMODYNAMIC SUB-REGION 3 AND 4
%
C=[-1.72260420D-2 -7.77175039D+0 4.20460752D+0 -2.76807038D+0 ...
      2.10419707D+0 -1.14649588D+0 2.23138085D-1 1.16250363D-1 ...
      -8.20900544D-2 0.D+0 7.08636085D-1 1.23679455D+1 -1.20389004D+1 ...
      5.40437422D+0 -9.93865043D-1 6.27523182D-2 -7.74743016D+0 0.D+0 ...
      0.D+0 0.D+0 -4.29885092D+0 4.31430538D+1 -1.41619313D+1 ...
      4.04172459D+0 1.55546326D+0 -1.66568935D+0 3.24881158D-1 ...
      2.93655325D+1 0.D+0 0.D+0 7.94841842D-6 8.0885947D+1 ...
      -8.36153380D+1 3.58636517D+1 7.51895954D+0 -1.26160640D+1 ...
      1.09717462D+0 2.12145492D+0 -5.46529566D-1 2.75971776D-6 ...
      -5.09073985D-4 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      0.D+0 2.10636332D+2  0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
      0.D+0 0.D+0 0.D+0 5.528935335D-2 -2.336365955D-1 3.697071420D-1 ...
      -2.596415470D-1 6.828087013D-2];
```

```
C010=1.94129239D-2;
C011=-1.69470576D-3;
C012=-4.311577033D+0;
C310=8.32875413D+0;
D=[0.D+0 0.D+0 0.D+0 0.D+0 0.D+0; ...
      0.D+0 0.D+0 3.526389875D+0 -2.642777743D+0 -1.236521258D-3; ...
      0.D+0 0.D+0 -2.690899373D+0 1.996765362D+0 1.155018309D-3; ...
      0.D+0 0.D+0 9.070982605D-1 -6.661557013D-1 0.D+0; ...
      0.D+0 0.D+0 -1.138791156D-1 8.270860589D-2 0.D+0];
D=transpose(D);
%
%     PSEUDOCRITICAL DENSITY OF HEAVY WATER TO BE USED IN THE
%     CALCULATION OF THERMAL CONDUCTIVITY AND VISCOCITY
%
PSEUDODC=0.358;
```

II.1.4 Function densf (αρχείο densf.m)

```
function[DF,IER]=densf
%
%     COMPUTE DENSITY OF LIQUID HEAVY WATER IN g/cm3 GIVEN PRESSURE
%     IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
%     IS EQUAL TO 0 SATURATED LIQUID DENSITY IS RETURNED
%
IER=0;
constants;
transient;
EPS=1.D-5; NDEC=5; ITMAX=100;
DF=0; DL=0;
[DL,IER]=svlwl;
f='fp';
df='dfpd';
[DF,IER]=newton(f,df,EPS,NDEC,DL,ITMAX);
ZZ=DF;
qmust(ZZ);
return
end
```

II.1.5 Function densg (αρχείο densg.m)

```
function[DG,IER]=densg
%
%     COMPUTE DENSITY OF HEAVY WATER VAPOR IN g/cm3 GIVEN PRESSURE
%     IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
%     IS EQUAL TO 0 SATURATED VAPOR DENSITY IS RETURNED
%
IER=0;
constants;
transient;
EPS=1.D-5; NDEC=5;ITMAX=100;
DG=0; DV=0;
[DV, IER]=svlwv;
FP='fp';
DFPD='dfpd';
[DG, IER]=newton(FP,DFPD,EPS,NDEC,DV,ITMAX);
DD=DG;
ZZ=DG;
qmust(ZZ);
return
end
```

II.1.6 Function dfpd (αρχείο dfpd.m)

```
function[DFPD]=dfpd(ZZ)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF HEAVY WATER PRESSURE BY
%     DENSITY
%
constants;
transient;
DFPD=R*TT*(1+2*ZZ*Q+4*ZZ*ZZ*QDT+ZZ*ZZ*ZZ*Q2D2T);
return
end
```

II.1.7 Function dfpr3 (αρχείο dfpr3.m)

```
function[DFPR3]=dfpr3(X3)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIGHT
%     WATER VAPOR BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
%     SUB-REGIONS 3 AND 5
%
constants;
transient;
DXSC0N=0.;
DXSC1N=0.;
DXSC2N=0.;
DXSC3N=0.;
DXSC4N=0.;
DXSC6N=0.;
for N=0:9
      if N>=2
            DXSC0N=DXSC0N+(N^2-N)*C(N)/(X3^(N+1));
      end
      if N>=2 & N<=6
            DXSC1N=DXSC1N+(N^2-N)*C(10+N)/(X3^(N+1));
      end
      if N>=2 & N<=7
            DXSC2N=DXSC2N+(N^2-N)*C(20+N)/(X3^(N+1));
      end
      if N>=2
            DXSC3N=DXSC3N+(N^2-N)*C(30+N)/(X3^(N+1));
      end
      if N<=4
            DXSC6N=DXSC6N+C(60+N)/(TR^(N+2));
      end
end
DXSC0N=DXSC0N+(100-10)*C010/(X3^11);
DXSC0N=DXSC0N+(121-11)*C011/(X3^12);
DXSC0N=DXSC0N-C012/(X3^2);
DXSC1N=DXSC1N-C(17)/(X3^2);
DXSC2N=DXSC2N-C(28)/(X3^2);
DXSC3N=DXSC3N-C310/(X3^2);
DXSC4N=-30*C(41)*(TR-1)/((X3^7)*(TR^23));
DXSC6N=30*(X3^4)*DXSC6N;
DFPR3=-DXSC0N-DXSC1N*(TR-1)-DXSC2N*((TR-1)^2)...
      -DXSC3N*((TR-1)^3)+DXSC4N-DXSC6N;
return
end
```

II.1.8 Function dfpr4 (αρχείο dfpr4.m)

```
function[DFPR4]=dfpr4(X4)
%
%     COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIQUID
%     LIGHT WATER BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
%     SUB-REGIONS 4 AND 5
%
constants;
transient;
Y=(1-TR)/(1-TR1);
DXSDN1=0.;
DXSDN2=0.;
DXSDMN=0.;
for M=3:4
      for N=1:5
            DXSDN1=DXSDN1-N*(N-1)*D(M,N)*(Y^M)/(X4^(N+1));
      end
            DXSDMN=DXSDMN+DXSDN1;
end
for N=1:3
      DXSDN2=DXSDN2+(N-1)*(N-2)*D(5,N)*(X4^(N-3));
end
DXSDN2=(Y^32)*DXSDN2;
[DFPR3]=dfpr3(X4);
DFPR4=DFPR3+DXSDMN-DXSDN2;
return
end
```

II.1.9 Function dfps.m (αρχείο dfps.m)

```
function[DFPS]=dfps(TS)
%
%     COMPUTE 1st DERIVATIVE OF HEAVY WATER SATURATION PRESSURE BY
%     TEMPERATURE
%
constants;
transient;
TAUS=1-(TS/TC);
FPSS=A1*TAUS+A2*(TAUS^1.9D+0)+A4*TAUS*TAUS+A11*(TAUS^5.5D+0)+ ...
      A20*(TAUS^10);
DFPS=(PC/10)*exp((TC/TS)*FPSS)*((-TC/(TS*TS))*FPSS+ ...
      (-1/TS)*(A1+A2*1.9D+0*(TAUS^0.9D+0)+A4*2*TAUS+A11*5.5D+0*...
      (TAUS^4.5D+0)+A20*10*(TAUS^9)));
return
end
```

II.1.10 Function energy (αρχείο energy.m)

```
function[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD)
%
%     COMPUTE SPECIFIC INTERNAL ENERGY OF HEAVY WATER IN kJ/kg GIVEN
%     TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
U=PSI0-TT*PSI0T+R*1000*DD*QTD;
PSI=PSI0+R*TT*(log(DD)+DD*Q);
return
end
```

II.1.11 Function enthalpy (αρχείο enthalpy.m)

```
function[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD)
%
%     COMPUTE ENTHALPY OF HEAVY WATER IN kJ/kg GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
H=R*TT*(1+DD*Q+DD*(1000/TT)*QTD+DD*DD*QDT)+PSI0-TT*PSI0T;
return
end
```

II.1.12 Function entropy (αρχείο entropy.m)

```
function[S]=entropy(TT,DD,PSI0T,Q,QTD)
%
%     COMPUTE ENTROPY OF HEAVY WATER IN kJ/(kgK) GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
S=-R*(log(DD)+DD*Q-DD*(1000/TT)*QTD)-PSI0T;
return
end
```

II.1.13 Script example.m

```
'program example';
%
%MASTER TEST PROGRAM FOR HEAVY WATER PROPERTY PACKAGE
%
constants;
transient;
disp('National Technical University of Athens')
disp('Department of Mechanical Engineering')
disp('Nuclear Engineering Section')
disp(' ')
disp('HEAVY W.A.S.P. (R)')
disp('HEAVY WATER AND STEAM PROPERTIES PACKAGE')
disp('Thermodynamic and Transport Properties')
disp('of the Heavy Water Substance;an')
disp('interactive approach')
disp('MATLAB Version 1-SEPTEMBER 2010')
disp('Author: STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>')
disp('This version of code accepts')
disp('as input only pressure and tempreture')
disp('(range: 0.006601-1000bar,3.8-500 C)')
disp('To obtain a saturation state, equal either')
disp('pressure or temperature to zero(for the')
disp('desired temperature or pressure,respectively')
disp(' ')
SUPER=false;
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
%
[PP PR TT TR DD J]=deal(0.);
disp(' ')
disp('Please enter desired pressure in bars:')
P=input('P:');
disp('Please enter desired temperature in C:')
T=input('T:');
if T~=0.
      T=T+273.15;
end
JS=1;
JP=63;
[P,T,TS,SVF,SVG,JR,IS]=heavy_wasp(JS,JP,P,T);
if JR==0
      return
end
fprintf('Pressure in bars:%8.3f\n',P);
fprintf('Temperature in C:%8.3f\n',T-273.15);
disp(' ')
if P>PC&T>TC
      SUPER=true;
      disp('SUPERCRITICAL STATE ')
end
if P<=PC&SVF~=0&IS~=2
      disp('SUBCOOLED LIQUID REGION ')
end
if P<=PC&SVG~=0&IS~=2
      disp('SUPERHEATED STEAM REGION ')
end
if IS==0
      disp('There is no saturation state for this pressure and temperature')
      disp(' ')
```

```
      disp(' ')
end
if IS==1
      disp('Exact saturation temperature for this pressure')
      fprintf('%7.3f\n',TS-273.15);
      disp(' ')
end
if IS==2
      disp('This is a saturation state or almost a saturation state')
      disp(' ')
      disp(' ')
      disp(' ')
end
if SUPER == true
      disp('DENSE PHASE ')
if SVF>0.
      disp(' ')
      fprintf('Specific Volume in m3/kg:%8.7f\n',SVF);
      fprintf('Internal Energy in KJ/Kg:%6.1f\n',UF);
      fprintf('Specific Enthalpy in KJ/Kg:%6.1f\n',HF);
      fprintf('Specific Entropy in KJ/KgK:%6.4f\n',SF);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVF);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUF);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      fprintf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
      disp(' ')
      return
end
if SVG>0.
      disp(' ')
      fprintf('Specific Volume in m3/kg:%10.6f\n',SVG);
      fprintf('Internal Energy in kJ/Kg:%6.1f\n',UG);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      fprintf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUG);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      fprintf('Prandtl number:%6.3f\n',PRANDTLG);
      disp(' ')
      disp(' ')
      return
end
end
      disp('LIQUID ')
      disp(' ')
      fprintf('Specific Volume in m3/kg:%8.7f\n',SVF);
      fprintf('Internal Energy in kJ/kg:%6.1f\n',UF);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HF);
      fprintf('Specific Entropy in kJ/kgK:%6.4f\n',SF);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
      fprintf('Specific Heat atconstant volume in kJ/(kgK):%6.4f\n',CVF);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUF);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
      fprintf('Prandtl number:%6.3f\n',PRANDTLF);
      disp(' ')
      disp('VAPOR')
      disp(' ')
      fprintf('Specific Volume in m3/kg:%10.6f\n',SVG);
```

```
      fprintf('Internal Energy in kJ/kg:%6.1f\n',UG);
      fprintf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
      fprintf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
      fprintf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
      fprintf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
      fprintf('Viscosity in kg/(ms):%8.7f\n',MUG);
      fprintf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
      fprintf('Prandtl number:%6.3f\n',PRANDTLG);
      disp(' ')
      disp(' ')
      disp(' ')
if IS==2
      fprintf('Surface tension in N/m:%7.6f\n',SIGMAFG);
else
      return
end
```

II.1.14 Function fp (αρχείο fp.m)

```
function[FP]=fp(ZZ)
%
%     COMPUTE PRESSURE OF HEAVY WATER IN MPa GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%
constants;
transient;
qmust(ZZ);
%
%     CONVERT PRESSURE IN BARS TO PRESSURE IN MPa
%
PPP=PP/10.;
FP=ZZ*R*TT*(1+ZZ*Q+ZZ*ZZ*QDT)-PPP;
return
end
```

II.1.15 Function fpr3 (αρχείο fpr3.m)

```
function[FPR3]=fpr3(X3)
%
%     COMPUTE REDUCED PRESSURE OF LIGHT WATER VAPOR IN THERMODYNAMIC
%     SUB-REGIONS 3 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
%     INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
%
constants;
transient;
XSC0N=0.;
XSC1N=0.;
XSC2N=0.;
XSC3N=0.;
XSC4N=0.;
XSC6N=0.;
for  N=0:9
      if N>=2
            XSC0N=XSC0N+(1-N)*C(N)/(X3^N);
      end
      if N>=2 & N<=6
            XSC1N=XSC1N+(1-N)*C(10+N)/(X3^N);
      end
      if N>=2 & N<=7
            XSC2N=XSC2N+(1-N)*C(20+N)/(X3^N);
      end
      if N>=2
            XSC3N=XSC3N+(1-N)*C(30+N)/(X3^N);
      end
      if N<=4
        XSC6N=XSC6N+C(60+N)/(TR^(N+2));
      end
end
XSC0N=XSC0N+(1-10)*C010/(X3^10);
XSC0N=XSC0N+(1-11)*C011/(X3^11);
XSC0N=XSC0N+C(1)+C012/X3;
XSC1N=XSC1N+C(11)+C(17)/X3;
XSC2N=XSC2N+C(21)+C(28)/X3;
XSC3N=XSC3N+C(31)+C310/X3;
XSC4N=5*C(41)*(TR-1)/((X3^6)*(TR^23));
XSC6N=6*(X3^5)*XSC6N;
FPR3=-XSC0N-XSC1N*(TR-1)-XSC2N*((TR-1)^2)-XSC3N*((TR-1)^3) ...
      +XSC4N-XSC6N-PR;
```

```
return
end
```

II.1.16 Function fpr4 (αρχείο fpr4.m)

```
function[FPR4]=fpr4(X4)
%
%     COMPUTE REDUCED PRESSURE OF LIQUID LIGHT WATER IN THERMODYNAMIC
%     SUB-REGIONS 4 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
%     INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
%
constants;
transient;
Y=(1-TR)/(1-TR1);
XSDN1=0.;
XSDN2=0.;
XSDMN=0.;
for M=3:4
      for N=1:5
            XSDN1=XSDN1+(N-1)*D(M,N)*(Y^M)/(X4^N);
      end
      XSDMN=XSDMN+XSDN1;
end
for N=1:3
      XSDN2=XSDN2+(N-1)*D(5,N)*(X4^(N-2));
end
XSDN2=(Y^32)*XSDN2;
[FPR3]=fpr3(X4);
FPR4=FPR3+XSDMN-XSDN2;
return
end
```

II.1.17 Function fps (αρχείο fps.m)

```
function[FPS]=fps(TS)
%
%     COMPUTE SATURATION PRESSURE OF HEAVY WATER IN MPa AS
%     FUNCTION OF TEMPERATURE IN K UNITS
%
constants;
transient;
%
%     CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
%
PPC=PC/10.;
PPP=PP/10.;
TAUS=1-(TS/TC);
FPS=PPC*exp((TC/TS)*(A1*TAUS+A2*(TAUS^1.9)+ ...
      A4*TAUS*TAUS+A11*(TAUS^5.5)+A20*(TAUS^10.)))-PPP;
return
end
```

II.1.18 Function fts (αρχείο fts.m)

```
function[FTS]=fts
%
%     COMPUTE FIRST APPROXIMATION OF HEAVY WATER SATURATION
%     TEMPERATURE IN K UNITS AS FUNCTION OF PRESSURE IN BARS
%     ACCORDING TO AUTHORS FIT.THIS APPROXIMATION IS TO BE USED
%     AS INITIAL GUESS FOR NUMERICAL CALCULATION OF ACTUAL SATURATION
%     TEMPERATURE OF HEAVY WATER
%
constants;
transient;
A0=5.44785913381933D-3;
B0=5.05259142381575D-2;
A1=7.59500801393874D-3;
B1=5.05259142381575D-2;
A2=3.89933736564079D-9;
B2=4.174981224613024D+0;
A3=2.367804557875880;
B3=1.21812837989753D-2;
if PP<0.006698D+0
      FTS=log(PP/A0)/B0+273.15D+0;
end
if PP>=0.006698D+0 & PP<1.D+0
      FTS=log(PP/A1)/B1+273.15D+0 ;
end
if PP>=1.D+0 & PP<180.D+0
      FTS=exp(log(PP/A2)/B2)+273.15D+0;
end
if PP>=180.D+0
      FTS=log(PP/A3)/B3+273.15D+0;
end
return
end
```

II.1.19 Function heavy_wasp (αρχείο heavy_wasp.m)

```
function [P,T,TS,SVF,SVG,JR,IS] = heavy_wasp(JS,JP,P,T)
IER=0;
constants;
transient;
%
%     INITIALIZE VALUES OF DENSITIES AND SPECIFIC VOLUMES
%
[DF DG SVF SVG]=deal(0.);
%
%     INITIALIZE VALUES OF OTHER PROPERTIES
%
initial;
%
%     HEAVY WATER AND STEAM PROPERTY PACKAGE DEVELOPED BY
%     STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>
%
%     COMPUTE THERMODYNAMIC AND TRANSPORT PROPERTIES OF HEAVY WATER
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN DEGREES CENTIGRADE
%     MAINLY ACCORDING TO EQUATION OF STATE DEVELOPED BY HILL P.G.,
%     MACMILLAN R.D.C. AND LEE V.,AND EQUATIONS FOR TRANSPORT
%     PROPERTIES DEVELOPED BY MATSUNAGA N. AND NAGASHIMA A.
%     PARTS OF CODE STRUCTURE ARE BASED TO SIMILAR PROPERTY PACKAGE
%     FOR LIGHT WATER DEVELOPED BY HENDRICKS R.C.,PELLER I.C. AND
%     BARON A.K.PARTS OF EQUATION OF STATE FOR LIGHT WATER DEVELOPED
%     BY SCHMIDT E. AND GRIGULL U. ARE USED TO PROVIDE GOOD INITIAL
%     GUESSES FOR NUMERICAL CALCULATION OF HEAVY WATER DENSITY.
%     VALUE OF JS(INPUT) MUST BE 1.CALCULATED THERMODYNAMIC AND
%     TRANSPORT PROPERTIES ARE SPECIFIED BY JP(INPUT).JR(OUTPUT)
%     REPRESENTS THERMODYNAMIC SUB-REGION OF P-T DIAGRAMM IN WHICH
%     INPUT LIES.IS(OUTPUT) INDICATES RELATIVE POSITION OF INPUT TO
%     SATURATION CURVE.COMPUTE SATURATION PROPERTIES IF EITHER PRESSURE
%     OR TEMPERATURE IS EQUAL TO ZERO (FOR RESPECTIVE TEMPERATURE OR
%     PRESSURE)
%
%     CHECK FOR CORRECT CALL OF HEAVY_WASP ROUTINE
%
if JS~=1
      disp('Internal Error: Illegal JS parameter')
      return
end
if JP>63 | JP<0
      disp('Internal Error: Illegal JS parameter')
      return
end
PP=P;
TT=T;
TS=0.;
%
%     CHECK FOR OUT OF RANGE
%
[TS,JR,IS]=checkpt;
if JR==0
      return
end
P=PP;
T=TT;
JJ=JR;
%     DENSITY OF LIQUID HEAVY WATER
%
```

```
if JJ==1 | JJ==6 | JJ==4 | JJ==5
      [DF,IER]=densf;
      if DF~=0.
%
%     CONVERT DENSITY TO SPECIFIC VOLUME IN m3/Kg
%
      SVF=1/(1000*DF);
      DD=DF;
      end
%
      %CHECK FOR CRITICAL POINT VICINITY
%
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have no validity!')
      return
      end
%
%     OTHER PROPERTIES OF LIQUID HEAVY WATER
%
      if JP~=0
      [B2F,B3F,B4F,DB2F,DB3F,DB4F,UF,PSIF,HF,SF,CPF,...
      CVF,IBMF,JTCF,GAMMAF,AF,MUF,KF,PRANDTLF,SIGMAFG]=total(JP,JJ,...
      TT,TR,DD,PSI0,PSI0T,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
      end
end
%
%     DENSITY OF HEAVY WATER VAPOR
%
if JJ==2 | JJ==6 | JJ==3 | JJ==5
      [DG,IER]=densg;
      if DG~=0.
%
%     CONVERT DENSITY TO SPECIFIC VOLUME IN m3/kg
%
      SVG=1/(1000*DG);
      DD=DG;
      end
%
%     CHECK FOR CRITICAL POINT VICINITY
%
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have no validity!')
      return
      end
%
%     OTHER PROPERTIES OF HEAVY WATER VAPOR
%
      if JP~=0
      [B2G,B3G,B4G,DB2G,DB3G,DB4G,UG,PSIG,HG,SG,CPG,...
      CVG,IBMG,JTCG,GAMMAG,AG,MUG,KG,PRANDTLG,SIGMAFG]=total(JP,JJ,...
      TT,TR,DD,PSI0,PSI0T,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
      end
end
if JJ==5 | JJ==6
%
%     COMPUTE LATENT HEAT OF HEAVY WATER IN kJ/kg
%
      HFG=HG-HF;
%
%     COMPUTE LAPLACE CONSTANT OF HEAVY WATER IN m
%
```

```
      [DG,IER]=densg;
      [DF,IER]=densf;
      if DF~=0. & DG~=0. & abs(DF-DG)>=1.D-3
      G=9.80665D+0;
      LACFG =sqrt(SIGMAFG/(G*1000*(DF-DG)));
      end
end
disp(' ')
disp(' ')
disp(' ')
return
end
```

II.1.20 Function ibm_jtc (αρχείο ibm_jtc.m)

```
function[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE ISOTHERMAL BULK MODULUS IN 1/MPa AND JOULE-THOMSON
%     COEFFICIENT IN K/MPa
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
JTC=(1/(DD*CP))*((TT/DD)*(PTD/PDT)-1);
IBM=1./(DD*PDT);
return
end
```

II.1.21 Script initial.m

```
'INITIAL';
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
%     REGARDLESS OF PHASE
%
[D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC]=deal(0.);
[GAMMA A MU K PRANDTL SIGMA]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY WATER
%     SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
%
[DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF JTCF]=deal(0.);
[GAMMAF AF MUF KF PRANDTLF]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER VAPOR
%     SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
%
[DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG JTCG]=deal(0.);
[GAMMAG AG MUG KG PRANDTLG]=deal(0.);
%
%     VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN SATURATION
%     STATES,SUFFIX FG STANDS FOR SATURATION
%
[HFG SIGMAFG LACFG]=deal(0.);
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE MAIN
%     DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE SERIES
%     EXPANSION OF Q
%
[Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2]=deal(0.);
```

II.1.22 Function newton (αρχείο newton.m)

```
function[X,IER]=newton(f,df,EPS,NDEC,X,ITMAX)
%
%     FIND REAL ZERO X OF REAL FUNCTION F GIVEN GOOD INITIAL GUESS
%     X AND FUNCTION DF (1st PARTIAL DERIVATIVE OF F BY X).ROOT X
%     IS ACCEPTED IF ABSOLUTE VALUE OF F(X).LE.EPS(INPUT).ROOT X
%     IS ACCEPTED IF TWO SUCCESSIVE APPROXIMATIONS AGREE TO FIRST
%     NDEC(INPUT) DECIMAL DIGITS.X(INPUT) IS AN INITIAL VALUE.X(OUTPUT)
%     CONTAINS COMPUTED ROOT.ITMAX(INPUT) IS MAXIMUM ALLOWBLE NUMBER
%     OF NEWTON_RAPHSON ITERATIONS USED ON ROOT.IER(OUTPUT) WARNING
%     ERROR PARAMETER=N,N=1 ROOT X WAS BYPASSED BECAUSE DF BECOMES
%     TOO SMALL.X IS SET TO 111111.THIS ERROR CONDITION MAY CAUSE AN
%     OVERFLOW,N=2 ROOT X WAS BYPASSED  BECAUSE ITMAX WAS EXCEEDED.
%     X IS SET TO 222222,N=3 SEVERAL OF THE ABOVE ERROR CONDITIONS
%     OCCURED.X IS SET TO EITHER 111111. OR 222222. AS ABOVE.DOUBLE
%     PRECISION ROUTINE.
%
TOL=10.^(-NDEC);
IER=0;
if ITMAX>100
      ITMAX=100;
end
for  K=1:ITMAX
      dd=feval(df,X);
      if abs(dd)<EPS
            X=111111.D+0;
            IER=1;
            return
      end
      ff=feval(f,X);
      dd=feval(df,X);
      XK=X-ff/dd;
      if K==41
            TOL=TOL*10;
      end
      if K==61
      TOL=TOL*10;
      end
      if K==81
      TOL=TOL*10 ;
      end
      if abs(XK-X)<TOL
            X=XK;
            return
      else
            X=XK;
      ff=feval(f,X);
      if abs(ff)<EPS
      return
      end
      end
end
      X=222222.D+0;
      if IER==0
      IER=2;
      else
      IER=3;
      end
return
end
```

II.1.23 Function prl (αρχείο prl.m)

```
function[PRL]=prl
%
%     COMPUTE REDUCED PRESSURE OF LIGHT WATER CORRESPONDING TO
%     REDUCED TEMPERATURE OF HEAVY WATER CALCULATED FROM GIVEN
%     TEMPERATURE IN K UNITS ALONG THE BOUNDARY BETWEEN THERMODYNAMIC
%     SUB-REGIONS 2 AND 3
%
constants;
transient;
L=7.160997524D+0;
PRL=((TR2-TR)*PR1+(TR-TR1)*PR2-L*(TR2-TR)*(TR-TR1))/(TR2-TR1);
return
end
```

II.1.24 Function prs (αρχείο prs.m)

```
function[PRS]=prs
%
%     COMPUTE REDUCED SATURATION VAPOR PRESSURE OF HEAVY WATER AS
%     FUNCTION OF TEMPERATURE IN K UNITS
%
constants;
transient;
TS=TR*TC;
TAUS=1-(TS/TC);
PRS=exp((TC/TS)*(A1*TAUS+A2*(TAUS^1.9)+ ...
      A4*TAUS*TAUS+A11*(TAUS^5.5)+A20*(TAUS^10.)));
return
end
```

II.1.25 Function qmust (αρχείο qmust.m)

```
function[]=qmust(ZZ)
constants;
transient;
%
%     MAIN DATA-FITTING SUBROUTINE
%
%     COEFFICIENTS Aij
%
A=[73.13848592D+0,-285.20415917D+0,535.71659288D+0,-
649.81000614D+0,...
574.63280680D+0,-387.92157774D+0,206.34569512D+0, -79.89428513D+0,...
-996.36169097D+0,-766.27290006D+0;...
24.74108348D+0,-105.57317181D+0,200.87302906D+0,-235.18776440D+0,...
224.56976938D+0,-40.09924297D+0,128.77154771D+0,-28.40907978D+0,...
-1389.08003142D+0,-1672.09705556D+0;...
11.64775625D+0,-42.51820251D+0,72.45541064D+0,-82.55391089D+0,...
0.D+0,0.D+0,0.D+0,0.D+0,-267.85482520D+0,-998.64982710D+0;...
2.66566642D+0,-9.19657655D+0,15.13096920D+0,-7.24860975D+0,...
0.D+0,0.D+0,0.D+0,0.D+0,-46.83904320D+0,-227.34793319D+0;
-6.73408249D+0,24.03602093D+0,-41.08079830D+0,45.39111005D+0...
0.D+0,0.D+0,0.D+0,0.D+0,139.21659329D+0,566.02305152D+0;...
-5.24802962D+0,18.52690633D+0,-31.42397369D+0,26.43208802D+0,...
0.D+0,0.D+0,0.D+0,0.D+0,96.31411481D+0,453.20280933D+0;...
-1.17583447D+0,4.13816432D+0,-6.55842224D+0,4.75774631D+0,...
0.D+0,0.D+0,0.D+0,0.D+0,19.39184297D+0,103.56819758D+0];
A=transpose(A);
%
%     COEFFICIENTS Ci
%
C=[1866.73D+0 4661.9D+0 64.605D+0 -284.8833D+0 100.1333D+0 -13.135D+0 ...
      0.32684D+0 -1211.253D+0];
%
%     OTHER CONSTANTS
%
TAJ1=1.553D+0;
TAJ2=2.53D+0;
E=4.3D+0;
RAJ1=0.7D+0;
RAJ2=1.1D+0;
%
%     TEMPERATURE PARAMETER
%
TAU=1000/TT;
TAUC=TAJ1;
EX=exp(-E*ZZ);
[SUMI1 SUMI3 SUMI5]=deal(0.);
for I=1:8
SUMI1=SUMI1+A(I,1)*(ZZ-RAJ1)^(I-1);
      if I>=2
            SUMI3=SUMI3+(I-1)*A(I,1)*(ZZ-RAJ1)^(I-2);
      end
      if I>=3
            SUMI5=SUMI5+A(I,1)*(I-1)*(I-2)*(ZZ-RAJ1)^(I-3);
      end
end
[SUMJ1 SUMJ2 SUMJ3 SUMJ4 SUMJ5 SUMJ6]=deal(0.);
for J=1:7
[SUMI2 SUMI4 SUMI6]=deal(0.);
      for I=1:8
```

```
      SUMI2=SUMI2+A(I,J)*(ZZ-RAJ2)^(I-1);
      if I>=2
            SUMI4=SUMI4+(I-1)*A(I,J)*(ZZ-RAJ2)^(I-2);
      end
      if I>=3
            SUMI6=SUMI6+A(I,J)*(I-1)*(I-2)*(ZZ-RAJ2)^(I-3);
      end
      end
SUMI2=SUMI2+EX*(A(9,J)+A(10,J)*ZZ);
SUMI4=SUMI4+EX*(-E*(A(9,J)+A(10,J)*ZZ)+A(10,J));
SUMI6=SUMI6+EX*(-E)*(-E*A(9,J)+A(10,J)*(2-E*ZZ));
      if J>=2
            SUMJ1=SUMJ1+((TAU-TAJ2)^(J-2))*SUMI2;
      end
      if J>=3
            SUMJ2=SUMJ2+(J-2)*((TAU-TAJ2)^(J-3))*SUMI2;
      end
      if J>=2
      SUMJ3=SUMJ3+((TAU-TAJ2)^(J-2))*SUMI4;
      end
      if J>=3
            SUMJ4=SUMJ4+(J-2)*((TAU-TAJ2)^(J-3))*SUMI4;
      end
      if J>=2
            SUMJ5=SUMJ5+((TAU-TAJ2)^(J-2))*SUMI6;
      end
      if J>=4
            SUMJ6=SUMJ6+(J-2)*(J-3)*((TAU-TAJ2)^(J-4))*SUMI2;
      end
end
SUMI1=SUMI1+EX*(A(9,1)+A(10,1)*ZZ);
SUMI3=SUMI3+EX*(-E*(A(9,1)+A(10,1)*ZZ)+A(10,1));
SUMI5=SUMI5+EX*(-E)*(-E*A(9,1)+A(10,1)*(2-E*ZZ));
%
%     DATA-FITTING FUNCTION,Q
%
Q=SUMI1+(TAU-TAUC)*SUMJ1;
%
%     1st PARTIAL DERIVATIVE OF Q BY TEMPERATURE
%
QTD=SUMJ1+(TAU-TAUC)*SUMJ2;
%
%     1st PARTIAL DERIVATIVE OF Q BY DENSITY
%
QDT=SUMI3+(TAU-TAUC)*SUMJ3;
%
%     2nd PARTIAL DERIVATIVE OF Q BY DENSITY AND TEMPERATURE
%
Q2DT=SUMJ3+(TAU-TAUC)*SUMJ4;
%
%     2nd PARTIAL DERIVATIVE OF Q BY DENSITY
%
Q2D2T=SUMI5+(TAU-TAUC)*SUMJ5;
%
%     2nd PARTIAL DERIVATIVE OF Q BY TEMPERATURE
%
Q2T2D=2*SUMJ2+(TAU-TAUC)*SUMJ6;
SUMI7=0.D+0;
SUMI8=0.D+0;
SUMI9=0.D+0 ;
for I=1:6
```

```
SUMI7=SUMI7+C(I)*((TT/1000.)^(I-1));
SUMI8=SUMI8+C(I)*(I-1)*((TT/1000.)^(I-2));
SUMI9=SUMI9+C(I)*(I-1)*(I-2)*((TT/1000.)^(I-3));
end
%
%     REFERENCE FUNCTION,PSI0
%
PSI0=SUMI7+C(7)*log(TT)+C(8)*TT*log(TT)/1000;
%
%     1st PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
%
PSI0T=SUMI8/1000+C(7)/TT+C(8)*log(TT)/1000+C(8)/1000;
%
%     2nd PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
%
PSI02T2=SUMI9*(1.D-6)-C(7)/(TT*TT)+C(8)/(1000*TT);
return
end
```

II.1.26 Function shp (αρχείο shp.m)

```
function[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE SPECIFIC HEAT OF HEAVY WATER AT CONSTANT PRESSURE IN
%     kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
      PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
            *Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
return
end
```

II.1.27 Function shr_sove (αρχείο shr_sove.m)

```
function[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     COMPUTE ISENTROPIC EXPANSION COEFFICIENT(SPECIFIC HEAT RATIO)
%     AND SONIC VELOCITY IN m/s OF HEAVY WATER
%
constants;
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
%
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
%
%     1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
%
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
%
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)*...
      Q2DT+DD*DD*QTD));
%
%     1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
%
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
%
%     COMPUTE CP
%
CP=HTD-HDT*PTD/PDT;
%
%     COMPUTE CV
%
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
GAMMA=CP/CV;
A=(10^(3/2.))*(1./sqrt(1/PDT-(TT/(CP*(DD^2)))*(PTD^2)/...
      (PDT^2)));
return
end
```

II.1.28 Function shv (αρχείο shv.m)

```
function[CV]=shv(TT,DD,PSI02T2,Q2T2D)
%
%     COMPUTE SPECIFIC HEAT OF HEAVY WATER AT CONSTANT VOLUME IN
%     kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
%
constants;
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
return
end
```

II.1.29 Function svlwl (αρχείο svlwl.m)

```
function[DL,IER]=svlwl
%
%     COMPUTE DENSITY OF LIQUID LIGHT WATER IN g/cm3 CORRESPONDING TO
%     REDUCED PRESSURE AND TEMPERATURE OF HEAVY WATER CALCULATED FROM
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
%
IER=0;
constants;
transient;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS IN THERMODYNAMIC
%     SUB-REGION 1
%
AM=[8.438375405D-1 5.362162162D-4 1.72D+0 7.342278489D-2 4.97585887D-2 ...
6.5371543D-1 1.15D-6 1.5108D-5 1.4188D-1 7.002753165D+0 ...
2.995284926D-4 2.04D-1];
A=[0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
7.982692717D+0 -2.616571843D-2 1.52241179D-3 2.284279054D-2 ...
2.421647003D+2 1.269716088D-10 2.074838328D-7 2.17402035D-8 ...
1.105710498D-9 1.293441934D+1 1.308119072D-5 6.047626338D-14];
 EPS=1.D-5; NDEC=5; ITMAX=100;
%
%     ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
if JJ==6 | JJ==5
      [PRS]=prs;
      PR=PRS;
end
Y=1-(AM(1)*TR*TR)-(AM(2)/(TR^6));
Z=AM(3)*Y*Y-2*AM(4)*TR+2*AM(5)*PR;
if Z<=0.
      Z=Y;
else
      Z=Y+sqrt(Z);
end
      X1=A(11)*AM(5)/(Z^(5./17.))...
      +(A(12)+A(13)*TR+A(14)*TR*TR+A(15)*((AM(6)-TR)^10)...
      +A(16)/(AM(7)+(TR^19)))...
      -(A(17)+2*A(18)*PR+3*A(19)*PR*PR)/(AM(8)+(TR^11))...
      -A(20)*(TR^18)*(AM(9)+TR*TR)...
      *(-3/((AM(10)+PR)^4)+AM(11))...
      +3*A(21)*(AM(12)-TR)*PR*PR+4*(A(22)/(TR^20))*(PR^3);
if JJ==4
      X4=X1;
      f='fpr4';
      df='dfpr4';
      [X4,IER]=newton(f,df,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
if JJ==5
      X4=X1;
      f='fpr4';
      df='dfpr4';
      [X4,IER]=newton(f,df,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
PR=PP/PC;
VL=X1*VC;
DL=1/VL;
```

```
return
end
```

II.1.30 Function svlwv (αρχείο svlwl.m)

```
function[DV,IER]=svlwv
%
%     COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 CORRESPONDING TO
%     REDUCED PRESSURE AND TEMPERATURE OF HEAVY WATER CALCULATED FROM
%     GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
%
IER=0;
constants;
transient;
%
%     NUMERICAL VALUES OF PRIMARY CONSTANTS
%
%
%     THERMODYNAMIC SUB-REGION 2
%
B0=7.633333333D-1;
B90=1.936587558D+2;
B=[6.670375918D-2 1.388983801D+0 0.D+0 0.D+0 0.D+0 0.D+0;...
      8.390104328D-2 2.614670893D-2 -3.373439453D-2 0.D+0 0.D+0 0.D+0;...
      4.520918904D-1 1.069036614D-1 0.D+0 0.D+0 0.D+0 0.D+0 ;...
      -5.975336707D-1 -8.847535804D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      5.958051609D-1 -5.159303373D-1 2.075021122D-1 0.D+0 0.D+0 0.D+0;...
      1.190610271D-1 -9.867174132D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      1.683998803D-1 -5.809438001D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      6.552390126D-3 5.710218649D-4 0.D+0 0.D+0 0.D+0 0.D+0;...
      -1.388522425D+3 4.126607219D+3 -6.508211677D+3 5.745984054D+3 ...
      -2.693088365D+3 5.235718623D+2];
BM=[];
BM(6,1)=4.006073948D-1;
BM(7,1)=8.636081627D-2;
BM(8,1)=-8.532322921D-1;
BM(8,2)=3.460208861D-1;
%
%     NUMBERS OF TERMS n(m) AND l(m),AND EXPONENTS z(m,n) AND x(m,n)
%
NM=[2 3 2 2 3 2 2 2];
ZM=[13 3 0;...
      18 2 1;...
      18 10 0;...
      25 14 0;...
      32 28 24;...
      12 11 0;...
      24 18 0;...
      24 14 0];
LM=[0 0 0 0 0 1 1 2];
XM=[0 0;...
      0 0;...
      0 0;...
      0 0;...
      14 0;...
      19 0;...
      54 27];
ESP=1.D-5; NDEC=5; ITMAX=100;
%
%     ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
%
if JJ==6 | JJ==5
      [PRS]=prs;
```

```
      PR=PRS;
end
I1=10*R*TC/(PC*VC);
X=exp(B0*(1-TR));
VSM1=0.D+0;
VSM2=0.D+0;
VSN=0.D+0;
for M=1:8
      VSN1=0.D+0;
      VSN2=0.D+0;
      VSL=0.D+0;
      if M<=5
            for N=1:NM(M)
                  VSN1=VSN1+B(M,N)*(X^ZM(M,N));
            end
                  VSM1=VSM1+M*(PR^(M-1))*VSN1;
      else
      for N=1:NM(M)
            VSN2=VSN2+B(M,N)*(X^ZM(M,N));
      end
      for L=1:LM(M)
            VSL=VSL+BM(M,L)*(X^XM(M,L));
      end
            VSM2=VSM2+((M-2)*(PR^(1-M))*VSN2)/(((PR^(2-M))+VSL)^2);
      end
end
[PRL]=prl;
VSN=VSN+(11.*(PR/PRL)^10)*B90;
for N=1:6
      [PRL]=prl;
      VSN=VSN+(11.*(PR/PRL)^10)*B(9,N)*(X^N);
end
X2=I1*(TR/PR)-VSM1-VSM2+VSN;
if JJ==3 | JJ==5
      if X2<=0
            X3=I1*(TR/PR);
      else
      X3=X2;
      end
      F='fpr3';
      DF='dfpr3';
      [X3,IER]=newton(F,DF,ESP,NDEC,X3,ITMAX);
      X2=X3;
end
PR=PP/PC;
VV=VC*X2;
DV=1/VV;
return
end
```

II.1.31 Function tension (αρχείο tension.m)

```
function[SIGMA]=tension(TT,JJ)
%
%     COMPUTE SURFACE TENSION OF HEAVY WATER IN N/m GIVEN
%     TEMPERATURE IN K UNITS ACCORDING TO EQUATION DEVELOPED BY
%     STRAUB J.,ROSNER N. AND GRIGULL U.
%
constants;
SIGMA0=245.3D-3; M=1.27D+0; B=-0.663D+0;
TCSTR=644.65D+0;
SIGMA=0;
if JJ~=6 & JJ~=5
      return
end
TAUT=1-TT/TCSTR;
SIGMA=SIGMA0*(TAUT^M)*(1+B*TAUT);
return
end
```

II.1.32 Function total (αρχείο total.m)

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
      A,MU,K,PRANDTL,SIGMA]=total(JP,JJ,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
      Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
%
%     MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
%     IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
%     MUST BE CALLED TWICE.
%


%
%     ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
%     SPECIFICATION,JP
%
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 42 43 46 ...
      47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44 45 46 ...
      47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44 45 ...
      46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51 52 53 ...
      54 55 56 57 58 59 60 61 62 63];
[D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC]=deal(0.);
[GAMMA A MU K PRANDTL SIGMA]=deal(0.);
if mod(JP,2)<0
      T60;
end
if mod(JP,2)==0
      T70;
      T100;
end
if mod(JP,2)>0
      T60;
end
%
%     VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
%     TEMPERATURE AND INTERNAL ENERGY
%
function[]=T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70;
return
end
function[]=T70
for I=1:32
      if JP-JPC1(I)<0
      T100;
      return
      end
      if JP-JPC1(I)==0
      T90;
      return
      end
end
return
end
%
%     ENTHALPY AND ENTROPY
%
```

```
function[]=T90
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
T100;
return
end
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120;
      return
      end
end
return
end
%
%     SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT
VOLUME,
%     SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
%     MODULUS AND JOULE-THOMSON COEFFICIENT
%
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130;
return
end
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150;
      return
      end
end
return
end
%
%     VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
%
function[]=T150
[MU]=viscosity(TR,DD);
[K]=conductive(TR,DD);
if K~=0.D+0
      PRANDTL=1000.*MU*CP/K;
end
T160;
return
end
function[]=T160
for I=1:32
      if JP-JPC4(I)<0
```

```
      T190;
      return
      end
      if JP-JPC4(I)==0
      T180;
      return
      end
end
return
end
%
%     surface tension
%
function[]=T180
if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
end
T190;
return
end
function[]=T190
      if JP-32<0
      T200;
      return
      end
      if JP-32==0
      T210;
      return
      end
      if JP-32>0
      T210;
      return
      end
return
end
      function[]=T200
      return
      end
      function[]=T210
      return
      end
end
```

II.1.33 Script transient.m

```
'TRANSIENT';
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
%
global PP PR TT TR DD JJ

global D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC
global GAMMA A MU K PRANDTL
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY WATER
%     SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
%
global DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF JTCF
global GAMMAF AF MUF KF PRANDTLF
%
%     VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER VAPOR
%     SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
%
global DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG JTCG
global GAMMAG AG MUG KG PRANDTLG
%
%     VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN SATURATION
%     STATES,SUFFIX FG STANDS FOR SATURATION
%
global HFG SIGMAFG LACFG
%
%     VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE MAIN
%     DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE SERIES
%     EXPANSION OF Q
global Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2
```

II.1.34 Function viscosity (αρχείο viscosity.m)

```
function[MU]=viscosity(TR,DD)
%
%     COMPUTE VISCOSITY OF HEAVY WATER IN kg/(ms) GIVEN TEMPERATURE
%     IN K UNITS AND DENSITY IN g/cm3
%     COMPUTATION AT VICINITY OF CRITICAL POINT IS NOT ACCURATE WITH
%     THIS FIT
%
constants;
A=[0.4864192D+0 0.3509007D+0 -0.2847572D+0 0.07013759D+0
0.01641220D+0 ...
-0.01163815D+0 0.D+0;-0.2448372D+0 1.315436D+0 -1.037026D+0 ...
0.4660127D+0 -0.02884911D+0 -0.008239587D+0 0.D+0;...
-0.8702035D+0 1.297752D+0 -1.287846D+0 0.2292075D+0 0.D+0 0.D+0 ...
0.D+0;0.8716056D+0 1.353448D+0 0.D+0 -0.4857462D+0 0.1607171D+0 0.D+0 ...
-0.003886659D+0;-1.051126D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0;...
0.3458395D+0 0.D+0 -0.02148229D+0 0.D+0 -0.009603846D+0 ...
0.004559914D+0 0.D+0];
B=[1.00000D+0 0.940695D+0 0.578377D+0 -0.202044D+0];
HETA=55.26516D-6;
PSEUDODR=DD/PSEUDODC;
NSIJ=0.;
NSK=0.;
for I=1:6
      for J=1:7
      NSIJ=NSIJ+A(I,J)*(((1./TR)-1)^(I-1))*((PSEUDODR-1)^(J-1));
      end
end
for K=1:4
NSK=NSK+B(K)*((1./TR)^(K-1));
end
N0=HETA*(TR^0.5)*(1./NSK);
N=N0*exp(PSEUDODR*NSIJ);
MU=N;
return
end
```

ΙΙ.2 Περιβάλλον προγραμματισμού SCILAB

Τα υποπρογράμματα παρατίθενται κατά την αλφαβητική σειρά της ονομασίας τους. Συμπεριλαμβάνεται το κυρίως πρόγραμμα example.sci.

ΙΙ.2.1 Function checkpt (αρχείο checkpt.sci)

```
function[TS,JR,IS]=checkpt
IER=0;
TS=0.;
JR=0;
IS=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
EPS=1.D-5; NDEC=5; ITMAX=100;
//
//    CHECK PRESSURE FOR OUT OF RANGE
//
if (PP<PMIN | PP>PMAX) & (PP~=0.)
      disp(' ')
      disp('Desired Pressure out of range')
      disp(' ')
      return
end
//
//    CHECK TEMPERATURE FOR OUT OF RANGE
//
if (TT<TMIN | TT>TMAX) & (TT~=0.)
      disp(' ')
      disp('Desired Temperature out of range')
      disp(' ')
      return
end
//
//    IF PRESSURE AND TEMPERATURE ARE EQUAL TO 0 CHECK FOR CORRECT CALL
//    OF SATURATION PROPERTIES
//
if PP==0. & TT==0.
      disp(' ')
      disp('Both pressure and temperature cannot be 0.!')
      disp(' ')
      return
end
//
//    IF PRESSURE OR TEMPERATURE IS EQUAL TO 0 CHECK FOR OUT OF
//    SATURATION RANGE
//
if (PP>PC & TT==0.) | (TT>TC & PP==0.)
      disp(' ')
      disp('Such a saturation state does not exist!')
      disp(' ')
      return
end
if TT==0.;
      TT=TT+273.15D+0;
end
if PP==0.;
      [FPS]=fps(TT);
      PP=FPS*10;
```

```
      TS=TT;
      IS=1;
else
if PP<=PC & TT<=TC
      [FTS]=fts();
      TS=FTS;
      [FPS]=fps(TS);
      if abs(FPS)>EPS
            [TS,IER]=newton(fps,dfps,EPS,NDEC,TS,ITMAX);
            IS=1;
            if TT==273.15D+0
                  TT=TS;
            end
      end
end
end
if abs(TT-TS)<=1.D+0
      IS=2;
end
//
//    REDUCED PRESSURE AND TEMPERATURE
//
PR=PP/PC;
TR=TT/TC;
//
//    DETERMINE THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
[PRS]=prs();
[PRL]=prl();
if TR>=TRT & TR<=TR1
      if abs(TT-TS)<=1.D-5
            JR=6;
            return
      else
      if PR>=0. & PR<PRS
            JR=2;
            return
      end
      if PR>PRS & PR<=PR2
            JR=1;
            return
      end
      end
end
if TR>=TR1 & ((TR-1.)<1.D-5)
      if abs(TT-TS)<=1.D-5
            JR=5;
            return
      end
else
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<PRS
            JR=3;
            return
      end
      if PR>PRS & PR<PR2
            JR=4;
            return
```

```
      end
end
if (abs(TR-1))>=1.D-5 & TR<TR2
      if PR>=0. & PR<=PRL
            JR=2;
            return
      end
      if PR>PRL & PR<=PR2
            JR=3;
            return
      end
end
if TR>=TR2 & TR<=TR3
      JR=2;
      return
end
if JR==0
      disp(' ')
      disp('Internal Error: JR indicator value is zero')
      disp(' ')
      return
end
return
endfunction
```

II.2.2 Function conductive (αρχείο conductive.sci)

```
function[KK]=conductive(TR,DD)
//
//    COMPUTE THERMAL CONDUCTIVITY OF HEAVY WATER IN W/(mK) GIVEN
//    TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//    COMPUTATION AT VICINITY OF CRITICAL POINT IS NOT ACCURATE WITH
//    THIS FIT
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
A0=1.00000D+0;
A=[37.3223D+0 22.5485D+0 13.0465D+0 0.D+0 -2.60735D+0];
B=[483.65600D+0 -191.039D+0 73.0358D+0 -7.57467D+0];
BE=-2.506D+0;
B0=-167.31D+0;
CT1=0.144847D+0;
CT2=-5.64493D+0;
CR1=-2.80000D+0;
CR2=-0.080738543D+0;
CR3=-17.9430D+0;
PSEUDODR1=0.125698D+0;
C1=0.354296D+5;
C2=0.5D+10;
D1=-741.112D+0;
LAMDA=0.742128D-3;
PSEUDODR=DD/PSEUDODC;
EL0=A0;
for I=1:5
      EL0=EL0+A(I)*(TR^I);
end
DELTAEL=0.D+0;
for J=1:4
      DELTAEL=DELTAEL+B(J)*(PSEUDODR^J);
end
DELTAEL=DELTAEL+B0*(1-exp(BE*PSEUDODR));
F1=exp(CT1*TR+CT2*(TR^2));
F2=exp(CR1*((PSEUDODR-1)^2))+CR2*exp(CR3*((PSEUDODR-PSEUDODR1)^2));
TAUR=TR/(abs(TR-1.1)+1.1);
F3=1+exp(60*(TAUR-1)+20);
F4=1+exp(100*(TAUR-1)+15);
DELTAELC=1+(F2^2)*(C2*(F1^4)/F3+3.5*F2/F4);
DELTAELC=C1*F1*F2*DELTAELC;
DELTAELL=D1*(F1^1.2)*(1-exp(-((PSEUDODR/2.5)^10)));
EL=LAMDA*(EL0+DELTAEL+DELTAELC+DELTAELL);
KK=EL;
return
endfunction
```

II.2.3 Script constants.sci

```
"CONSTANTS";
//
//    DATA PROGRAM CONTAINING ALL GLOBALLY NEEDED CONSTANTS OF HEAVY_WASP
//    BEING USED AT LEAST TWICE. ALL OTHER ONCE USED CONSTANTS ARE
//    PART OF CORRESPONDING VALUE ASSIGNING COMMANDS IN SUBPROGRAMS.
//    TO COMMUNICATE WITH REST OF PACKAGE USE 'CONSTANTS' GLOBAL VARIABLE
//    DETERMINATION IN MAIN PROGRAM AND FUNCTIONS IF NECESSARY
//
global PMIN PMAX TMIN TMAX PC TC VC TRT TR1 TR2 TR3 PR1 PR2 R A1 A2 A4 A11
global A20 K  C C010 C011 C012 C310 D PSEUDODC JPC1 JPC2 JPC3 JPC4
//
//    PRESSURE AND TEMPERATURE RANGE OF EQUATIONS FOR THERMODYNAMIC AND
//    TRANSPORT PROPERTIES OF HEAVY WATER
//
PMIN=0.006601D+0; PMAX=1000.D+0; TMIN=276.95D+0; TMAX=775.D+0;
//
//    CRITICAL PRESSURE TEMPERATURE AND SPECIFIC VOLUME OF HEAVY WATER
//
PC=216.6D+0; TC=643.89D+0; VC=2.793D+0;
//
//    REDUCED TEMPERATURE RANGES OF THERMODYNAMIC SUB REGIONS
//
TRT=4.30120051D-1; TR1=9.626911787D-1;
TR2=1.333462073D+0; TR3=1.657886606D+0;
//
//    MAXIMUM REDUCED PRESSURE
//
PR1=7.454108957D-1; PR2=4.616805171D+0;
//
//    HEAVY WATER CONSTANT
//
R=0.415147D+0;
//
//    CONSTANTS OF HEAVY WATER SATURATION LINE
//
A1=-7.81583D+0;
A2=17.6012D+0;
A4=-18.1747D+0;
A11=-3.92488D+0;
A20=4.19174D+0;
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS OF EQUATION OF STATE OF LIGHT
//    WATER BEING USED IN THIS PACKAGE
//
//    SATURATION LINE
//
K=[-7.691234564D+0 -2.608023696D+1 -1.681706546D+2 6.423285504D+1 ...
      -1.189646225D+2 4.167117320D+0 2.097506760D+1 1.D+9 6.D+0];
//
//    THERMODYNAMIC SUB-REGION 3 AND 4
//
C=[-1.72260420D-2 -7.77175039D+0 4.20460752D+0 -2.76807038D+0 ...
2.10419707D+0 -1.14649588D+0 2.23138085D-1 1.16250363D-1 ...
-8.20900544D-2 0.D+0 7.08636085D-1 1.23679455D+1 -1.20389004D+1 ...
5.40437422D+0 -9.93865043D-1 6.27523182D-2 -7.74743016D+0 0.D+0 0.D+0 ...
0.D+0 -4.29885092D+0 4.31430538D+1 -1.41619313D+1 4.04172459D+0 ...
1.55546326D+0 -1.66568935D+0 3.24881158D-1 2.93655325D+1 0.D+0 0.D+0 ...
7.94841842D-6 8.0885947D+1 -8.36153380D+1 3.58636517D+1 ...
7.51895954D+0 -1.26160640D+1 1.09717462D+0 2.12145492D+0 ...
```

```
-5.46529566D-1 2.75971776D-6 -5.09073985D-4 0.D+0 0.D+0 0.D+0 0.D+0 ...
0.D+0 0.D+0 0.D+0 0.D+0 2.10636332D+2 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 ...
0.D+0 0.D+0 0.D+0 0.D+0 5.528935335D-2 -2.336365955D-1 3.697071420D-1 ...
-2.596415470D-1 6.828087013D-2];
C010=1.94129239D-2;
C011=-1.69470576D-3;
C012=-4.311577033D+0;
C310=8.32875413D+0;
D=[0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
0.D+0,0.D+0,0.D+0,0.D+0,0.D+0 ;...
0.D+0,3.526389875D+0,-2.690899373D+0,9.070982605D-1,-1.138791156D-1;...
0.D+0,-2.642777743D+0,1.996765362D+0,-6.661557013D-1,8.270860589D-2;...
0.D+0,-1.236521258D-3,1.155018309D-3,0.D+0,0.D+0];
//
//    PSEUDOCRITICAL DENSITY OF HEAVY WATER TO BE USED IN THE CALCULATION
//    OF THERMAL CONDUCTIVITY AND VISCOCITY
//
PSEUDODC=0.358;
```

II.2.4 Function densf (αρχείο densf.sci)

```
function[DF,IER]=densf
//
//    COMPUTE DENSITY OF LIQUID HEAVY WATER IN g/cm3 GIVEN PRESSURE
//    IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
//    IS EQUAL TO 0 SATURATED LIQUID DENSITY IS RETURNED
//
IER=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
EPS=1.D-5; NDEC=5; ITMAX=100;
DF=0; DL=0;
[DL,IER]=svlwl();
[DF,IER]=newton(fp,dfpd,EPS,NDEC,DL,ITMAX);
ZZ=DF;
qmust(ZZ);
return
endfunction
```

II.2.5 Function densg (αρχείο deng.sci)

```
function[DG,IER]=densg
//
//    COMPUTE DENSITY OF HEAVY WATER VAPOR IN g/cm3 GIVEN PRESSURE
//    IN BARS AND TEMPERATURE IN K UNITS.IF PRESSURE OR TEMPERATURE
//    IS EQUAL TO 0 SATURATED VAPOR DENSITY IS RETURNED
//
IER=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
EPS=1.D-5; NDEC=5;ITMAX=100;
DG=0; DV=0;
[DV, IER]=svlwv();
[DG, IER]=newton(fp,dfpd,EPS,NDEC,DV,ITMAX);
DD=DG;
ZZ=DG;
qmust(ZZ);
return
endfunction
```

II.2.6 Function dfpd (αρχείο dfpd.sci)

```
function[DFPD]=dfpd(ZZ)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF HEAVY WATER PRESSURE BY
//    DENSITY
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
DFPD=R*TT*(1+2*ZZ*Q+4*ZZ*ZZ*QDT+ZZ*ZZ*ZZ*Q2D2T);
return
endfunction
```

II.2.7 Function dfpr3 (αρχείο dfpr3.sci)

```
function[DFPR3]=dfpr3(X3)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIGHT
//    WATER VAPOR BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
//    SUB-REGIONS 3 AND 5
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
DXSC0N=0.;
DXSC1N=0.;
DXSC2N=0.;
DXSC3N=0.;
DXSC4N=0.;
DXSC6N=0.;
for N=0:9
      if N>=2
            DXSC0N=DXSC0N+(N^2-N)*C(N)/(X3^(N+1));
      end
      if N>=2 & N<=6
            DXSC1N=DXSC1N+(N^2-N)*C(10+N)/(X3^(N+1));
      end
      if N>=2 & N<=7
            DXSC2N=DXSC2N+(N^2-N)*C(20+N)/(X3^(N+1));
      end
      if N>=2
            DXSC3N=DXSC3N+(N^2-N)*C(30+N)/(X3^(N+1));
      end
      if N<=4
            DXSC6N=DXSC6N+C(60+N)/(TR^(N+2));
      end
end
DXSC0N=DXSC0N+(100-10)*C010/(X3^11);
DXSC0N=DXSC0N+(121-11)*C011/(X3^12);
DXSC0N=DXSC0N-C012/(X3^2);
DXSC1N=DXSC1N-C(17)/(X3^2);
DXSC2N=DXSC2N-C(28)/(X3^2);
DXSC3N=DXSC3N-C310/(X3^2);
DXSC4N=-30*C(41)*(TR-1)/((X3^7)*(TR^23));
DXSC6N=30*(X3^4)*DXSC6N;
DFPR3=-DXSC0N-DXSC1N*(TR-1)-DXSC2N*((TR-1)^2)...
      -DXSC3N*((TR-1)^3)+DXSC4N-DXSC6N;
return
endfunction
```

II.2.8 Function dfpr4 (αρχείο dfpr4.sci)

```
function[DFPR4]=dfpr4(X4)
//
//    COMPUTE 1st PARTIAL DERIVATIVE OF REDUCED PRESSURE OF LIQUID
//    LIGHT WATER BY REDUCED SPECIFIC VOLUME IN THERMODYNAMIC
//    SUB-REGIONS 4 AND 5
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
Y=(1-TR)/(1-TR1);
DXSDN1=0.;
DXSDN2=0.;
DXSDMN=0.;
for M=3:4
      for N=1:5
            DXSDN1=DXSDN1-N*(N-1)*D(M,N)*(Y^M)/(X4^(N+1));
      end
      DXSDMN=DXSDMN+DXSDN1;
end
for N=1:3
      DXSDN2=DXSDN2+(N-1)*(N-2)*D(5,N)*(X4^(N-3));
end
DXSDN2=(Y^32)*DXSDN2;
[DFPR3]=dfpr3(X4);
DFPR4=DFPR3+DXSDMN-DXSDN2;
return
endfunction
```

II.2.9 Function dfps (αρχείο dfps.sci)

```
function[DFPS]=dfps(TS)
//
//    COMPUTE 1st DERIVATIVE OF HEAVY WATER SATURATION PRESSURE BY
//    TEMPERATURE
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
TAUS=1-(TS/TC);
FPSS=A1*TAUS+A2*(TAUS^1.9D+0)+A4*TAUS*TAUS+A11*(TAUS^5.5D+0)+ ...
      A20*(TAUS^10);
DFPS=(PC/10)*exp((TC/TS)*FPSS)*((-TC/(TS*TS))*FPSS+ ...
      (-1/TS)*(A1+A2*1.9D+0*(TAUS^0.9D+0)+A4*2*TAUS+A11*5.5D+0* ...
      (TAUS^4.5D+0)+A20*10*(TAUS^9)));
return
endfunction
```

II.2.10 Function energy (αρχείο energy.sci)

```
function[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD)
//
//    COMPUTE SPECIFIC INTERNAL ENERGY OF HEAVY WATER IN kJ/kg GIVEN
//    TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
U=PSI0-TT*PSI0T+R*1000*DD*QTD;
PSI=PSI0+R*TT*(log(DD)+DD*Q);
endfunction
```

II.2.11 Function enthalpy (αρχείο enthalpy.sci)

```
function[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD)
//
//    COMPUTE ENTHALPY OF HEAVY WATER IN kJ/kg GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
H=R*TT*(1+DD*Q+DD*(1000/TT)*QTD+DD*DD*QDT)+PSI0-TT*PSI0T;
return
endfunction
```

II.2.12 Function entropy (αρχείο entropy.sci)

```
function[S]=entropy(TT,DD,PSI0T,Q,QTD)
//
//    COMPUTE ENTROPY OF HEAVY WATER IN kJ/(kgK) GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
S=-R*(log(DD)+DD*Q-DD*(1000/TT)*QTD)-PSI0T;
return
endfunction
```

II.2.13 Script example.sci

```
"program example";
funcprot(0);
getd('F:\scilab_hw\functions');
//
//    MASTER TEST PROGRAM FOR HEAVY WATER PROPERTY PACKAGE
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
write(%io(2),'National Technical University of Athens')
write(%io(2),'Department of Mechanical Engineering')
write(%io(2),'Nuclear Engineering Section')
write(%io(2),' ')
write(%io(2),'HEAVY W.A.S.P. (R)')
write(%io(2),'HEAVY WATER AND STEAM PROPERTIES PACKAGE')
write(%io(2),'Thermodynamic and Transport Properties')
write(%io(2),'of the Heavy Water Substance;an')
write(%io(2),'interactive approach')
write(%io(2),'SCILAB Version 1-SEPTEMBER 2010')
write(%io(2),'Author: STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>')
write(%io(2),'This version of code accepts')
write(%io(2),'as input only pressure and tempreture')
write(%io(2),'(range: 0.006601-1000bar,3.8-500 C)')
write(%io(2),'To obtain a saturation state, equal either')
write(%io(2),'pressure or temperature to zero(for the')
write(%io(2),'desired temperature or pressure,respectively')
write(%io(2),' ')
SUPER=%f;
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
//
PP=0.; PR=0.; TT=0.; TR=0.; DD=0.; J=0.;
write(%io(2),' ')
write(%io(2),'Please enter desired pressure in bars:')
P=input('P:');
write(%io(2),'Please enter desired temperature in C:')
T=input('T:');
if T~=0
      T=T+273.15;
end
JS=1;
JP=63;
[P,T,TS,SVF,SVG,JR,IS]=heavy_wasp(JS,JP,P,T);
if JR==0
      return
end
printf('Pressure in bars:%8.3f\n',P);
printf('Temperature in C:%8.3f\n',T-273.15);
write(%io(2),' ')
if P>PC&T>TC
      SUPER=%t;
      write(%io(2),'SUPERCRITICAL STATE ')
end
if P<=PC&SVF~=0&IS~=2
      write(%io(2),'SUBCOOLED LIQUID REGION ')
end
if P<=PC&SVG~=0&IS~=2
      write(%io(2),'SUPERHEATED STEAM REGION ')
end
if IS==0
```

```
write(%io(2),'There is no saturation state for this pressure and ...
temperature')
write(%io(2),' ')
write(%io(2),' ')
end
if IS==1
write(%io(2),'Exact saturation temperature for this pressure')
printf('%7.3f\n',TS-273.15);
write(%io(2),' ')
end
if IS==2
      disp('This is a saturation state or almost a saturation state')
      disp(' ')
      disp(' ')
      disp(' ')
end
if SUPER==%t
      disp('DENSE PHASE ')
if SVF>0
      disp(' ')
printf('Specific Volume in m3/kg:%8.7f\n',SVF);
printf('Internal Energy in KJ/Kg:%6.1f\n',UF);
printf('Specific Enthalpy in KJ/Kg:%6.1f\n',HF);
printf('Specific Entropy in KJ/KgK:%6.4f\n',SF);
printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVF);
printf('Viscosity in kg/(ms):%8.7f\n',MUF);
printf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
printf('Prandtl number:%6.3f\n',PRANDTLF);
disp(' ')
disp(' ')
return
end
if SVG>0
disp(' ')
printf('Specific Volume in m3/kg:%10.6f\n',SVG);
printf('Internal Energy in kJ/Kg:%6.1f\n',UG);
printf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
printf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
printf('Viscosity in kg/(ms):%8.7f\n',MUG);
printf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
printf('Prandtl number:%6.3f\n',PRANDTLG);
disp(' ')
disp(' ')
return
end
end
disp('LIQUID ')
disp(' ')
printf('Specific Volume in m3/kg:%8.7f\n',SVF);
printf('Internal Energy in kJ/kg:%6.1f\n',UF);
printf('Specific Enthalpy in kJ/kg:%6.1f\n',HF);
printf('Specific Entropy in kJ/kgK:%6.4f\n',SF);
printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPF);
printf('Specific Heat atconstant volume in kJ/(kgK):%6.4f\n',CVF);
printf('Viscosity in kg/(ms):%8.7f\n',MUF);
printf('Thermal Conductivity in W/(mK):%5.4f\n',KF);
printf('Prandtl number:%6.3f\n',PRANDTLF);
disp(' ')
```

```
disp('VAPOR')
disp(' ')
printf('Specific Volume in m3/kg:%10.6f\n',SVG);
printf('Internal Energy in kJ/kg:%6.1f\n',UG);
printf('Specific Enthalpy in kJ/kg:%6.1f\n',HG);
printf('Specific Entropy in kJ/(kgK):%7.4f\n',SG);
printf('Specific Heat at constant pressure in kJ/(kgK):%10.4f\n',CPG);
printf('Specific Heat at constant volume in kJ/(kgK):%6.4f\n',CVG);
printf('Viscosity in kg/(ms):%8.7f\n',MUG);
printf('Thermal Conductivity in W/(mK):%5.4f\n',KG);
printf('Prandtl number:%6.3f\n',PRANDTLG);
disp(' ')
disp(' ')
disp(' ')
if IS==2
      disp('1')
      printf('Surface tension in N/m:%7.6f\n',SIGMAFG);
else
      return
end
```

II.2.14 Function fp (αρχείο fp.sci)

```
function[FP]=fp(ZZ)
//
//    COMPUTE PRESSURE OF HEAVY WATER IN MPa GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
qmust(ZZ);
//
//    CONVERT PRESSURE IN BARS TO PRESSURE IN MPa
//
PPP=PP/10.;
FP=ZZ*R*TT*(1+ZZ*Q+ZZ*ZZ*QDT)-PPP;
return
endfunction
```

II.2.15 Function fpr3 (αρχείο fpr3.sci)

```
function[FPR3]=fpr3(X3)
//
//    COMPUTE REDUCED PRESSURE OF LIGHT WATER VAPOR IN THERMODYNAMIC
//    SUB-REGIONS 3 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
//    INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
XSC0N=0.;
XSC1N=0.;
XSC2N=0.;
XSC3N=0.;
XSC4N=0.;
XSC6N=0.;
for  N=0:9
      if N>=2
            XSC0N=XSC0N+(1-N)*C(N)/(X3^N);
      end
      if N>=2 & N<=6
            XSC1N=XSC1N+(1-N)*C(10+N)/(X3^N);
      end
      if N>=2 & N<=7
            XSC2N=XSC2N+(1-N)*C(20+N)/(X3^N);
      end
      if N>=2
            XSC3N=XSC3N+(1-N)*C(30+N)/(X3^N);
      end
      if N<=4
            XSC6N=XSC6N+C(60+N)/(TR^(N+2));
      end
end
XSC0N=XSC0N+(1-10)*C010/(X3^10);
XSC0N=XSC0N+(1-11)*C011/(X3^11);
XSC0N=XSC0N+C(1)+C012/X3;
XSC1N=XSC1N+C(11)+C(17)/X3;
XSC2N=XSC2N+C(21)+C(28)/X3;
XSC3N=XSC3N+C(31)+C310/X3;
XSC4N=5*C(41)*(TR-1)/((X3^6)*(TR^23));
XSC6N=6*(X3^5)*XSC6N;
FPR3=-XSC0N-XSC1N*(TR-1)-XSC2N*((TR-1)^2)-XSC3N*((TR-1)^3) ...
      +XSC4N-XSC6N-PR;
```

```
return
endfunction
```

II.2.16 Function fpr4 (αρχείο fpr4.sci)

```
function[FPR4]=fpr4(X4)
//
//    COMPUTE REDUCED PRESSURE OF LIQUID LIGHT WATER IN THERMODYNAMIC
//    SUB-REGIONS 4 AND 5 CORRESPONDING TO REDUCED TEMPERATURE AND
//    INITIAL GUESS OF REDUCED SPECIFIC VOLUME OF HEAVY WATER
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
Y=(1-TR)/(1-TR1);
XSDN1=0.D+0;
XSDN2=0.D+0;
XSDMN=0.D+0;
for M=3:4
      for N=1:5
            XSDN1=XSDN1+(N-1)*D(M,N)*(Y^M)/(X4^N);
      end
      XSDMN=XSDMN+XSDN1;
end
for N=1:3
      XSDN2=XSDN2+(N-1)*D(5,N)*(X4^(N-2));
end
XSDN2=(Y^32)*XSDN2;
[FPR3]=fpr3(X4);
FPR4=FPR3+XSDMN-XSDN2;
return
endfunction
```

II.2.17 Function fps (αρχείο fps.sci)

```
function[FPS]=fps(TS)
//
//    COMPUTE SATURATION PRESSURE OF HEAVY WATER IN MPa AS
//    FUNCTION OF TEMPERATURE IN K UNITS
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    CONVERT PRESSURES IN BARS TO PRESSURES IN MPa
//
PPC=PC/10.;
PPP=PP/10.;
TAUS=1-(TS/TC);
FPS=PPC*exp((TC/TS)*(A1*TAUS+A2*(TAUS^1.9)+ ...
      A4*TAUS*TAUS+A11*(TAUS^5.5)+A20*(TAUS^10.)))-PPP;
return
endfunction
```

II.2.18 Function fts.sci

```
function[FTS]=fts
//
//    COMPUTE FIRST APPROXIMATION OF HEAVY WATER SATURATION
//    TEMPERATURE IN K UNITS AS FUNCTION OF PRESSURE IN BARS
//    ACCORDING TO AUTHORS FIT.THIS APPROXIMATION IS TO BE USED
//    AS INITIAL GUESS FOR NUMERICAL CALCULATION OF ACTUAL SATURATION
//    TEMPERATURE OF HEAVY WATER
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
A0=5.44785913381933D-3;
B0=5.05259142381575D-2;
A1=7.59500801393874D-3;
B1=5.05259142381575D-2;
A2=3.89933736564079D-9;
B2=4.174981224613024D+0;
A3=2.367804557875880;
B3=1.21812837989753D-2;
if PP<0.006698D+0
      FTS=log(PP/A0)/B0+273.15D+0;
end
if PP>=0.006698D+0 & PP<1.D+0
      FTS=log(PP/A1)/B1+273.15D+0 ;
end
if PP>=1.D+0 & PP<180.D+0
      FTS=exp(log(PP/A2)/B2)+273.15D+0;
end
if PP>=180.D+0
      FTS=log(PP/A3)/B3+273.15D+0;
end
return
endfunction
```

II.2.19 Function heavy_wasp.sci

```
function [P,T,TS,SVF,SVG,JR,IS] = heavy_wasp(JS,JP,P,T)
IER=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    INITIALIZE VALUES OF DENSITIES AND SPECIFIC VOLUMES
//
DF=0.; DG=0.; SVF=0.; SVG=0.;
//
//    INITIALIZE VALUES OF OTHER PROPERTIES
//
exec('F:\scilab_hw\initial.sci');
//
//    HEAVY WATER AND STEAM PROPERTY PACKAGE DEVELOPED BY
//    STYLIANOS PACHITSAS & NICK PETROPOULOS<C:2010>
//
//    COMPUTE THERMODYNAMIC AND TRANSPORT PROPERTIES OF HEAVY WATER
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN DEGREES CENTIGRADE
//    MAINLY ACCORDING TO EQUATION OF STATE DEVELOPED BY HILL P.G.,
//    MACMILLAN R.D.C. AND LEE V.,AND EQUATIONS FOR TRANSPORT
//    PROPERTIES DEVELOPED BY MATSUNAGA N. AND NAGASHIMA A.
//    PARTS OF CODE STRUCTURE ARE BASED TO SIMILAR PROPERTY PACKAGE
//    FOR LIGHT WATER DEVELOPED BY HENDRICKS R.C.,PELLER I.C. AND
//    BARON A.K.PARTS OF EQUATION OF STATE FOR LIGHT WATER DEVELOPED
//    BY SCHMIDT E. AND GRIGULL U. ARE USED TO PROVIDE GOOD INITIAL
//    GUESSES FOR NUMERICAL CALCULATION OF HEAVY WATER DENSITY.
//    VALUE OF JS(INPUT) MUST BE 1.CALCULATED THERMODYNAMIC AND
//    TRANSPORT PROPERTIES ARE SPECIFIED BY JP(INPUT).JR(OUTPUT)
//    REPRESENTS THERMODYNAMIC SUB-REGION OF P-T DIAGRAMM IN WHICH
//    INPUT LIES.IS(OUTPUT) INDICATES RELATIVE POSITION OF INPUT TO
//    SATURATION CURVE.COMPUTE SATURATION PROPERTIES IF EITHER PRESSURE
//    OR TEMPERATURE IS EQUAL TO ZERO (FOR RESPECTIVE TEMPERATURE OR
//    PRESSURE)
//
//    CHECK FOR CORRECT CALL OF HEAVY_WASP ROUTINE
//
if JS~=1
      disp('Internal Error: Illegal JS parameter')
      return
end

if JP>63 | JP<0
      disp('Internal Error: Illegal JS parameter')
      return
end
PP=P;
TT=T;
TS=0.;
//
//    CHECK FOR OUT OF RANGE
//
[TS,JR,IS]=checkpt();
if JR==0
      return
end
P=PP;
T=TT;
JJ=JR;
//
```

```
//    DENSITY OF LIQUID HEAVY WATER
//
if JJ==1 | JJ==6 | JJ==4 | JJ==5
      [DF,IER]=densf();
      if DF~=0.
//
//    CONVERT DENSITY TO SPECIFIC VOLUME IN m3/Kg
//
      SVF=1/(1000*DF);
      DD=DF;
      end
//
//    CHECK FOR CRITICAL POINT VICINITY
//
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have no validity!')
      return
      end
//
//    OTHER PROPERTIES OF LIQUID HEAVY WATER
//
if JP~=0
[B2F,B3F,B4F,DB2F,DB3F,DB4F,UF,PSIF,HF,SF,CPF,CVF,IBMF,JTCF,GAMMAF,...
AF,MUF,KF,PRANDTLF,SIGMAFG]=total(JP,J,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
end
end
//
//    DENSITY OF HEAVY WATER VAPOR
//
if JJ==2 | JJ==6 | JJ==3 | JJ==5
      [DG,IER]=densg();
      if DG~=0.
//
//    CONVERT DENSITY TO SPECIFIC VOLUME IN m3/kg
//
      SVG=1/(1000*DG);
      DD=DG;
      end
//
//    CHECK FOR CRITICAL POINT VICINITY
//
      if abs(T-TC)<=10 & abs(DD/PSEUDODC-1)<=0.3
      disp('Calculations near the critical point have no validity!')
      return
      end
//
//    OTHER PROPERTIES OF HEAVY WATER VAPOR
//
if JP~=0
[B2G,B3G,B4G,DB2G,DB3G,DB4G,UG,PSIG,HG,SG,CPG,CVG,IBMG,JTCG,GAMMAG,...
AG,MUG,KG,PRANDTLG,SIGMAFG]=total(JP,J,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
end
end
if JJ==5 | JJ==6
//
//    COMPUTE LATENT HEAT OF HEAVY WATER IN kJ/kg
//
      HFG=HG-HF;
//
```

```
//    COMPUTE LAPLACE CONSTANT OF HEAVY WATER IN m
//
      [DG,IER]=densg();
      [DF,IER]=densf();
      if DF~=0. & DG~=0. & abs(DF-DG)>=1.D-3
      G=9.80665D+0;
      LACFG=sqrt(SIGMAFG/(G*1000*(DF-DG)));
      end
end
disp(' ')
disp(' ')
disp(' ')
return
endfunction
```

II.2.20 Function ibm_jtc (αρχείο ibm_jtc.sci)

```
function[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE ISOTHERMAL BULK MODULUS IN 1/MPa AND JOULE-THOMMSON
//    COEFFICIENT IN K/MPa
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
      *Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
JTC=(1/(DD*CP))*((TT/DD)*(PTD/PDT)-1);
IBM=1./(DD*PDT);
return
endfunction
```

II.2.21 Script initial (αρχείο initial.sci)

```
"INITIAL";
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER
//    REGARDLESS OF PHASE
//
D=0.; B2=0.; B3=0.; B4=0.; DB2=0.; DB3=0.; DB4=0.;
U=0.; PSI=0.; H=0.; S=0.; CP=0.; CV=0.; IBM=0.; JTC=0.;
GAMMA=0.; A=0.; MU=0.; K=0.; PRANDTL=0.; SIGMA=0.;
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY WATER
//    SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
//
DF=0.; B2F=0.; B3F=0.; B4F=0.; DB2F=0.; DB3F=0.; DB4F=0.; UF=0.;
PSIF=0.; HF=0.; SF=0.; CPF=0.; CVF=0.; IBMF=0.; JTCF=0.;
GAMMAF=0.; AF=0.; MUF=0.; KF=0.; PRANDTLF=0.;
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER VAPOR
//    SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
//
DG=0.; B2G=0.; B3G=0.; B4G=0.; DB2G=0.; DB3G=0.; DB4G=0.;
UG=0.; PSIG=0.; HG=0.; SG=0.; CPG=0.; CVG=0.; IBMG=0.; JTCG=0.;
GAMMAG=0.; AG=0.; MUG=0.; KG=0.; PRANDTLG=0.;
//
//    VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN SATURATION
//    STATES, SUFFIX FG STANDS FOR SATURATION
//
HFG=0.; SIGMAFG=0.; LACFG=0.;
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE MAIN
//    DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE SERIES
//    EXPANSION OF Q
//
Q=0.; QDT=0.; Q2D2T=0.;
QTD=0.; Q2T2D=0.; Q2DT=0.; PSI0=0.; PSI0T=0.; PSI02T2=0.;
```

II.2.22 Function newton (αρχείο newton.sci)

```
function[X,IER]=newton(f,df,EPS,NDEC,X,ITMAX)
//
//    FIND REAL ZERO X OF REAL FUNCTION F GIVEN GOOD INITIAL GUESS
//    X AND FUNCTION DF (1st PARTIAL DERIVATIVE OF F BY X).ROOT X
//    IS ACCEPTED IF ABSOLUTE VALUE OF F(X).LE.EPS(INPUT).ROOT X
//    IS ACCEPTED IF TWO SUCCESSIVE APPROXIMATIONS AGREE TO FIRST
//    NDEC(INPUT) DECIMAL DIGITS.X(INPUT) IS AN INITIAL VALUE.X(OUTPUT)
//    CONTAINS COMPUTED ROOT.ITMAX(INPUT) IS MAXIMUM ALLOWBLE NUMBER
//    OF NEWTON_RAPHSON ITERATIONS USED ON ROOT.IER(OUTPUT) WARNING
//    ERROR PARAMETER=N,N=1 ROOT X WAS BYPASSED BECAUSE DF BECOMES
//    TOO SMALL.X IS SET TO 111111.THIS ERROR CONDITION MAY CAUSE AN
//    OVERFLOW,N=2 ROOT X WAS BYPASSED  BECAUSE ITMAX WAS EXCEEDED.
//    X IS SET TO 222222,N=3 SEVERAL OF THE ABOVE ERROR CONDITIONS
//    OCCURED.X IS SET TO EITHER 111111. OR 222222. AS ABOVE.DOUBLE
//    PRECISION ROUTINE.
//
TOL=10.^(-NDEC);
IER=0;
if ITMAX>100
      ITMAX=100;
end
for K=1:ITMAX
      dd=feval(X,df);
      if abs(dd)<EPS
            X=111111.D+0;
            IER=1;
            return
      end
      ff=feval(X,f);
      dd=feval(X,df);
      XK=X-ff/dd;
      if K==41
            TOL=TOL*10;
      end
      if K==61
            TOL=TOL*10;
      end
      if K==81
            TOL=TOL*10 ;
      end
      if abs(XK-X)<TOL
            X=XK;
            return
      else
            X=XK;
            ff=feval(X,f);
      if abs(ff)<EPS
            return
      end
      end
end
X=222222.D+0;
if IER==0
      IER=2;
else
      IER=3;
end
return
endfunction
```

II.2.23 Function prl (αρχείο prl.sci)

```
function[PRL]=prl
//
//    COMPUTE REDUCED PRESSURE OF LIGHT WATER CORRESPONDING TO
//    REDUCED TEMPERATURE OF HEAVY WATER CALCULATED FROM GIVEN
//    TEMPERATURE IN K UNITS ALONG THE BOUNDARY BETWEEN THERMODYNAMIC
//    SUB-REGIONS 2 AND 3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
L=7.160997524D+0;
PRL=((TR2-TR)*PR1+(TR-TR1)*PR2-L*(TR2-TR)*(TR-TR1))/(TR2-TR1);
return
endfunction
```

II.2.24 Function prs (αρχείο prs.sci)

```
function[PRS]=prs
//
//    COMPUTE REDUCED SATURATION VAPOR PRESSURE OF HEAVY WATER AS
//    FUNCTION OF TEMPERATURE IN K UNITS
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
TS=TR*TC;
TAUS=1-(TS/TC);
PRS=real(exp((TC/TS)*(A1*TAUS+A2*(TAUS^1.9)+ ...
      A4*TAUS*TAUS+A11*(TAUS^5.5)+A20*(TAUS^10.))));
return
endfunction
```

II.2.25 Function qmust (αρχείο qmust.sci)

```
function[]=qmust(ZZ)
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    MAIN DATA-FITTING SUBROUTINE
//
//    COEFFICIENTS Aij
//
A=[73.13848592D+0,24.74108348D+0,11.64775625D+0,2.66566642D+0,...
-6.73408249D+0,-5.24802962D+0,-1.17583447D+0;...
-285.20415917D+0,-105.57317181D+0,-42.51820251D+0,-
9.19657655D+0,24.03602093D+0,18.52690633D+0,4.13816432D+0;...
535.71659288D+0,200.87302906D+0,72.45541064D+0,15.13096920D+0,...
-41.08079830D+0,-31.42397369D+0,-6.55842224D+0;...
-649.81000614D+0,-235.18776440D+0,-82.55391089D+0,...
-7.24860975D+0,45.39111005D+0,26.43208802D+0,4.75774631D+0;...
574.63280680D+0,224.56976938D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
-387.92157774D+0,-40.09924297D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
206.34569512D+0,128.77154771D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
-79.89428513D+0,-28.40907978D+0,0.D+0,0.D+0,0.D+0,0.D+0,0.D+0;...
-996.36169097D+0,-1389.08003142D+0,-267.85482520D+0,...
-46.83904320D+0,139.21659329D+0,96.31411481D+0,19.39184297D+0;...
-766.27290006D+0,-1672.09705556D+0,-998.64982710D+0,...
-227.34793319D+0,566.02305152D+0,453.20280933D+0,103.56819758D+0];
//
//    COEFFICIENTS Ci
//
C=[1866.73D+0 4661.9D+0 64.605D+0 -284.8833D+0 100.1333D+0 -13.135D+0 ...
      0.32684D+0 -1211.253D+0];
//
//    OTHER CONSTANTS
//
TAJ1=1.553D+0;
TAJ2=2.53D+0;
E=4.3D+0;
RAJ1=0.7D+0;
RAJ2=1.1D+0;
//
//    TEMPERATURE PARAMETER
//
TAU=1000/TT;
TAUC=TAJ1;
EX=exp(-E*ZZ);
SUMI1=0.; SUMI3=0.; SUMI5=0.;
for I=1:8
SUMI1=SUMI1+A(I,1)*(ZZ-RAJ1)^(I-1);
      if I>=2
            SUMI3=SUMI3+(I-1)*A(I,1)*(ZZ-RAJ1)^(I-2);
      end
      if I>=3
            SUMI5=SUMI5+A(I,1)*(I-1)*(I-2)*(ZZ-RAJ1)^(I-3);
      end
end
SUMJ1=0.; SUMJ2=0.; SUMJ3=0.; SUMJ4=0.; SUMJ5=0.; SUMJ6=0.;
for J=1:7
      SUMI2=0.; SUMI4=0.; SUMI6=0.;
      for I=1:8
            SUMI2=SUMI2+A(I,J)*(ZZ-RAJ2)^(I-1);
            if I>=2
```

```
            SUMI4=SUMI4+(I-1)*A(I,J)*(ZZ-RAJ2)^(I-2);
            end
            if I>=3
                  SUMI6=SUMI6+A(I,J)*(I-1)*(I-2)*(ZZ-RAJ2)^(I-3);
            end
      end
      SUMI2=SUMI2+EX*(A(9,J)+A(10,J)*ZZ);
      SUMI4=SUMI4+EX*(-E*(A(9,J)+A(10,J)*ZZ)+A(10,J));
      SUMI6=SUMI6+EX*(-E)*(-E*A(9,J)+A(10,J)*(2-E*ZZ));
      if J>=2
            SUMJ1=SUMJ1+((TAU-TAJ2)^(J-2))*SUMI2;
      end
      if J>=3
            SUMJ2=SUMJ2+(J-2)*((TAU-TAJ2)^(J-3))*SUMI2;
      end
      if J>=2
            SUMJ3=SUMJ3+((TAU-TAJ2)^(J-2))*SUMI4;
      end
      if J>=3
            SUMJ4=SUMJ4+(J-2)*((TAU-TAJ2)^(J-3))*SUMI4;
      end
      if J>=2
            SUMJ5=SUMJ5+((TAU-TAJ2)^(J-2))*SUMI6;
      end
      if J>=4
            SUMJ6=SUMJ6+(J-2)*(J-3)*((TAU-TAJ2)^(J-4))*SUMI2;
      end
end
SUMI1=SUMI1+EX*(A(9,1)+A(10,1)*ZZ);
SUMI3=SUMI3+EX*(-E*(A(9,1)+A(10,1)*ZZ)+A(10,1));
SUMI5=SUMI5+EX*(-E)*(-E*A(9,1)+A(10,1)*(2-E*ZZ));
//
//    DATA-FITTING FUNCTION,Q
//
Q=SUMI1+(TAU-TAUC)*SUMJ1;
//
//    1st PARTIAL DERIVATIVE OF Q BY TEMPERATURE
//
QTD=SUMJ1+(TAU-TAUC)*SUMJ2;
//
//    1st PARTIAL DERIVATIVE OF Q BY DENSITY
//
QDT=SUMI3+(TAU-TAUC)*SUMJ3;
//
//    2nd PARTIAL DERIVATIVE OF Q BY DENSITY AND TEMPERATURE
//
Q2DT=SUMJ3+(TAU-TAUC)*SUMJ4;
//
//    2nd PARTIAL DERIVATIVE OF Q BY DENSITY
//
Q2D2T=SUMI5+(TAU-TAUC)*SUMJ5;
//
//    2nd PARTIAL DERIVATIVE OF Q BY TEMPERATURE
//
Q2T2D=2*SUMJ2+(TAU-TAUC)*SUMJ6;
SUMI7=0.D+0;
SUMI8=0.D+0;
SUMI9=0.D+0 ;
for I=1:6
      SUMI7=SUMI7+C(I)*((TT/1000.)^(I-1));
      SUMI8=SUMI8+C(I)*(I-1)*((TT/1000.)^(I-2));
```

```
      SUMI9=SUMI9+C(I)*(I-1)*(I-2)*((TT/1000.)^(I-3));
end
//
//    REFERENCE FUNCTION,PSI0
//
PSI0=SUMI7+C(7)*log(TT)+C(8)*TT*log(TT)/1000;
//
//    1st PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
//
PSI0T=SUMI8/1000+C(7)/TT+C(8)*log(TT)/1000+C(8)/1000;
//
//    2nd PARTIAL DERIVATIVE OF PSI0 BY TEMPERATURE
//
PSI02T2=SUMI9*(1.D-6)-C(7)/(TT*TT)+C(8)/(1000*TT);
return
endfunction
```

II.2.26 Function shp (αρχείο shp.sci)

```
function[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE SPECIFIC HEAT OF HEAVY WATER AT CONSTANT PRESSURE IN
//    kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
      PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)...
            *Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
CP=HTD-HDT*PTD/PDT;
return
endfunction
```

II.2.27 Function shr_sove (αρχείο shr_sove.sci)

```
function[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    COMPUTE ISENTROPIC EXPANSION COEFFICIENT(SPECIFIC HEAT RATIO)
//    AND SONIC VELOCITY IN m/s OF HEAVY WATER
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY TEMPERATURE
//
HTD=-TT*PSI02T2+R*(1+DD*Q+DD*DD*QDT-(1000/TT)...
      *(DD*QTD+DD*DD*Q2DT+DD*(1000/TT)*Q2T2D));
//
//    1st PARTIAL DERIVATIVE OF ENTHALPY BY DENSITY
//
HDT=TT*R*(Q+(1000/TT)*(QTD+DD*Q2DT)+3*DD*QDT+DD*DD*Q2D2T);
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY TEMPERATURE
//
PTD=R*(DD+DD*DD*Q+(DD^3)*QDT-(1000/TT)*((DD^3)*...
      Q2DT+DD*DD*QTD));
//
//    1st PARTIAL DERIVATIVE OF PRESSURE BY DENSITY
//
PDT=TT*R*(1+2*DD*Q+4*DD*DD*QDT+(DD^3)*Q2D2T);
//
//    COMPUTE CP
//
CP=HTD-HDT*PTD/PDT;
//
//    COMPUTE CV
//
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
GAMMA=CP/CV;
A=(10^(3/2.))*(1./sqrt(1/PDT-(TT/(CP*(DD^2)))*(PTD^2)/...
      (PDT^2)));
return
endfunction
```

II.2.28 Function shv (αρχείο shv.sci)

```
function[CV]=shv(TT,DD,PSI02T2,Q2T2D)
//
//    COMPUTE SPECIFIC HEAT OF HEAVY WATER AT CONSTANT VOLUME IN
//    kJ/(kgK) GIVEN TEMPERATURE IN K UNITS AND DENSITY IN g/cm3
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
CV=-R*DD*(1.D+6/(TT*TT))*Q2T2D-TT*PSI02T2;
return
endfunction
```

II.2.29 Function svlwl (αρχείο svlwl.sci)

```
function[DL,IER]=svlwl
//
//    COMPUTE DENSITY OF LIQUID LIGHT WATER IN g/cm3 CORRESPONDING TO
//    REDUCED PRESSURE AND TEMPERATURE OF HEAVY WATER CALCULATED FROM
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
//
IER=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS IN THERMODYNAMIC
//    SUB-REGION 1
//
AM=[8.438375405D-1 5.362162162D-4 1.72D+0 7.342278489D-2 4.97585887D-2 ...
      6.5371543D-1 1.15D-6 1.5108D-5 1.4188D-1 7.002753165D+0 ...
      2.995284926D-4 2.04D-1];
A=[0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0  0.D+0 ...
      7.982692717D+0 -2.616571843D-2 1.52241179D-3 2.284279054D-2 ...
      2.421647003D+2 1.269716088D-10 2.074838328D-7 2.17402035D-8 ...
      1.105710498D-9 1.293441934D+1 1.308119072D-5 6.047626338D-14];
EPS=1.D-5; NDEC=5; ITMAX=100;
//
//    ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
if JJ==6 | JJ==5
      [PRS]=prs();
      PR=PRS();
end
Y=1-(AM(1)*TR*TR)-(AM(2)/(TR^6));
Z=AM(3)*Y*Y-2*AM(4)*TR+2*AM(5)*PR;
if Z<=0
      Z=Y;
else
      Z=Y+sqrt(Z);
end
      X1=A(11)*AM(5)/(Z^(5./17.))...
      +(A(12)+A(13)*TR+A(14)*TR*TR+A(15)*((AM(6)-TR)^10)...
      +A(16)/(AM(7)+(TR^19)))...
      -(A(17)+2*A(18)*PR+3*A(19)*PR*PR)/(AM(8)+(TR^11))...
      -A(20)*(TR^18)*(AM(9)+TR*TR)...
      *(-3/((AM(10)+PR)^4)+AM(11))...
      +3*A(21)*(AM(12)-TR)*PR*PR+4*(A(22)/(TR^20))*(PR^3);
if JJ==4
      X4=X1;
      [X4,IER]=newton(fpr4,dfpr4,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
if JJ==5
      X4=X1;
      [X4,IER]=newton(fpr4,dfpr4,EPS,NDEC,X4,ITMAX);
      X1=X4;
end
PR=PP/PC;
VL=X1*VC;
DL=1/VL;
return
endfunction
```

II.2.30 Function svlwv (αρχείο svlwv.sci)

```
function[DV,IER]=svlwv
//
//    COMPUTE DENSITY OF LIGHT WATER VAPOR IN g/cm3 CORRESPONDING TO
//    REDUCED PRESSURE AND TEMPERATURE OF HEAVY WATER CALCULATED FROM
//    GIVEN PRESSURE IN BARS AND TEMPERATURE IN K UNITS
//
IER=0;
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
//
//    NUMERICAL VALUES OF PRIMARY CONSTANTS
//
//    THERMODYNAMIC SUB-REGION 2
//
B0=7.633333333D-1;
B90=1.936587558D+2;
B=[6.670375918D-2 1.388983801D+0 0.D+0 0.D+0 0.D+0 0.D+0;...
      8.390104328D-2 2.614670893D-2 -3.373439453D-2 0.D+0 0.D+0 0.D+0;...
      4.520918904D-1 1.069036614D-1 0.D+0 0.D+0 0.D+0 0.D+0 ;...
      -5.975336707D-1 -8.847535804D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      5.958051609D-1 -5.159303373D-1 2.075021122D-1 0.D+0 0.D+0 0.D+0;...
      1.190610271D-1 -9.867174132D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      1.683998803D-1 -5.809438001D-2 0.D+0 0.D+0 0.D+0 0.D+0;...
      6.552390126D-3 5.710218649D-4 0.D+0 0.D+0 0.D+0 0.D+0;...
      -1.388522425D+3 4.126607219D+3 -6.508211677D+3 5.745984054D+3 ...
      -2.693088365D+3 5.235718623D+2];
BM=[];
BM(6,1)=4.006073948D-1;
BM(7,1)=8.636081627D-2;
BM(8,1)=-8.532322921D-1;
BM(8,2)=3.460208861D-1;
//
//    NUMBERS OF TERMS n(m) AND l(m),AND EXPONENTS z(m,n) AND x(m,n)
//
NM=[2 3 2 2 3 2 2 2];
ZM=[13 3 0;...
      18 2 1;...
      18 10 0;...
      25 14 0;...
      32 28 24;...
      12 11 0;...
      24 18 0;...
      24 14 0];
LM=[0 0 0 0 0 1 1 2];
XM=[0 0;...
      0 0;...
      0 0;...
      0 0;...
      14 0;...
      19 0;...
      54 27];
ESP=1.D-5; NDEC=5; ITMAX=100;
//
//    ACT ACCORDING TO THERMODYNAMIC SUB-REGION SPECIFICATION,JR
//
if JJ==6 | JJ==5
      [PRS]=prs();
      PR=PRS();
```

```
end
I1=10*R*TC/(PC*VC);
X=exp(B0*(1-TR));
VSM1=0.D+0;
VSM2=0.D+0;
VSN=0.D+0;
for M=1:8
      VSN1=0.D+0;
      VSN2=0.D+0;
      VSL=0.D+0;
      if M<=5
            for N=1:NM(M)
            VSN1=VSN1+B(M,N)*(X^ZM(M,N));
            end
            VSM1=VSM1+M*(PR^(M-1))*VSN1;
      else
            for N=1:NM(M)
            VSN2=VSN2+B(M,N)*(X^ZM(M,N));
            end
            for L=1:LM(M)
            VSL=VSL+BM(M,L)*(X^XM(M,L));
            end
            VSM2=VSM2+((M-2)*(PR^(1-M))*VSN2)/(((PR^(2-M))+VSL)^2);
      end
end
[PRL]=prl();
VSN=VSN+(11.*(PR/PRL)^10)*B90;
for N=1:6
      [PRL]=prl();
      VSN=VSN+(11.*(PR/PRL)^10)*B(9,N)*(X^N);
end
X2=I1*(TR/PR)-VSM1-VSM2+VSN;
if JJ==3 | JJ==5
      if X2<=0
            X3=I1*(TR/PR);
      else
            X3=X2;
      end
      [X3,IER]=newton(fpr3,dfpr3,ESP,NDEC,X3,ITMAX);
      X2=X3;
end
PR=PP/PC;
VV=VC*X2;
DV=1/VV;
return
endfunction
```

II.2.31 Function tension (αρχείο tension.sci)

```
function[SIGMA]=tension(TT,JJ)
//
//    COMPUTE SURFACE TENSION OF HEAVY WATER IN N/m GIVEN
//    TEMPERATURE IN K UNITS ACCORDING TO EQUATION DEVELOPED BY
//    STRAUB J.,ROSNER N. AND GRIGULL U.
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
SIGMA0=245.3D-3; M=1.27D+0; B=-0.663D+0;
TCSTR=644.65D+0;
SIGMA=0;
if JJ~=6 & JJ~=5
      return
end
TAUT=1-TT/TCSTR;
SIGMA=SIGMA0*(TAUT^M)*(1+B*TAUT);
return
endfunction
```

II.2.32 Function total (αρχείο total.sci)

```
function[B2,B3,B4,DB2,DB3,DB4,U,PSI,H,S,CP,CV,IBM,JTC,GAMMA,...
A,MU,KK,PRANDTL,SIGMA]=total(JP,J,TT,TR,DD,PSI0,PSI0T,PSI02T2,...
Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D)
//
//    MASTER SERVICE SUBPROGRAM DIRECTLY UNDER HEAVY_WASP ROUTINE.
//    IF VALUES OF SATURATION PROPERTIES ARE NEEDED THIS SUBPROGRAM
//    MUST BE CALLED TWICE.
//
//    ACT ACCORDING TO THERMODYNAMIC AND TRANSPORT PROPERTIES
//    SPECIFICATION,JP
//
exec('F:\scilab_hw\transient.sci');
JPC1=[2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31 34 35 38 39 42 43 46 ...
      47 50 51 54 55 58 59 62 63];
JPC2=[4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31 36 37 38 39 44 45 46 ...
      47 52 53 54 55 60 61 62 63];
JPC3=[8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31 40 41 42 43 44 45 ...
      46 55 56 57 58 59 60 61 62 63];
JPC4=[16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 48 49 50 51 52 53 ...
      54 55 56 57 58 59 60 61 62 63];
D=0.; B2=0.; B3=0.; B4=0.; DB2=0.; DB3=0.;
DB4=0.; U=0.; PSI=0.; H=0.; S=0.; CP=0.;
CV=0.; IBM=0.;JTC=0.; GAMMA=0.; A=0.;
MU=0.; K=0.; PRANDTL=0.; SIGMA=0.;
//
//    VIRIAL COEFFICIENTS, 1st DERIVATIVES OF VIRIAL COEFFICIENTS BY
//    TEMPERATURE AND INTERNAL ENERGY
//
function T60
[B2,B3,B4,DB2,DB3,DB4]=virial(TT);
[U,PSI]=energy(TT,DD,PSI0,PSI0T,Q,QTD);
T70();
return
endfunction
function[]=T70
for I=1:32
      if JP-JPC1(I)<0
      T100();
      return
      end
      if JP-JPC1(I)==0
      T90();
      return
      end
end
return
endfunction
//
//    ENTHALPY AND ENTROPY
//
function[]=T90
[H]=enthalpy(TT,DD,PSI0,PSI0T,Q,QDT,QTD);
[S]=entropy(TT,DD,PSI0T,Q,QTD);
T100();
return
endfunction
function[]=T100
for I=1:32
      if JP-JPC2(I)<0
```

```
      T130;
      return
      end
      if JP-JPC2(I)==0
      T120();
      return
      end
end
return
endfunction
//
//    SPECIFIC HEAT AT CONSTANT PRESSURE,SPECIFIC HEAT AT CONSTANT VOLUME,
//    SPECIFIC HEAT RADIO,SONIC VELOCITY,ISOTHERMAL BULK
//    MODULUS AND JOULE-THOMSON COEFFICIENT
//
function[]=T120
[CP]=shp(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[CV]=shv(TT,DD,PSI02T2,Q2T2D);
[GAMMA,A]=shr_sove(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
[IBM,JTC]=ibm_jtc(TT,DD,PSI02T2,Q,QDT,QTD,Q2DT,Q2D2T,Q2T2D);
T130();
return
endfunction
function[]=T130
for I=1:32
      if JP-JPC3(I)<0
      T160;
      return
      end
      if JP-JPC3(I)==0
      T150();
      return
      end
end
return
endfunction
//
//    VISCOSITY, THERMAL CONDUCTIVITY AND PRANDTL NUMBER
//
function[]=T150
[MU]=viscosity(TR,DD);
[KK]=conductive(TR,DD);
if KK~=0.D+0
exec('F:\scilab_hw\transient.sci');
      PRANDTL=1000.*MU*CP/KK;
end
T160();
return
endfunction
function[]=T160
for I=1:32
      if JP-JPC4(I)<0
      T190();
      return
      end
      if JP-JPC4(I)==0
      T180();
      return
      end
end
return
```

```
endfunction
//
//    surface tension
//
function[]=T180
      if JJ==5 | JJ==6
      [SIGMA]=tension(TT,JJ);
      end
T190();
return
endfunction
function[]=T190
      if JP-32<0
      T200();
      return
      end
      if JP-32==0
      T210();
      return
      end
      if JP-32>0
      T210();
      return
      end
return
endfunction
      function[]=T200
      return
      endfunction
      function[]=T210
      return
      endfunction
      if pmodulo(JP,2)<0
      T60();
      end
      if pmodulo(JP,2)==0
      T70();
      T100();
      end
      if pmodulo(JP,2)>0
      T60();
      end
return
endfunction
```

II.2.33 Script transient.sci

```
"TRANSIENT";
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT INPUT OR OUTPUT
//
global PP PR TT TR DD JJ
global D B2 B3 B4 DB2 DB3 DB4 U PSI H S CP CV IBM JTC
global GAMMA A MU KK PRANDTL SIGMA
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF LIQUID HEAVY WATER
//    SUFFIX F STANDS MAINLY FOR FLUID(LIQUID
//
global DF B2F B3F B4F DB2F DB3F DB4F UF PSIF HF SF CPF CVF IBMF JTCF
global GAMMAF AF MUF KF PRANDTLF
//
//    VARIABLES BELOW REPRESENT MAINLY PROPERTIES OF HEAVY WATER VAPOR
//    SUFFIX G STANDS MAINLY FOR GAS(VAPOR)
//
global DG B2G B3G B4G DB2G DB3G DB4G UG PSIG HG SG CPG CVG IBMG JTCG
global GAMMAG AG MUG KG PRANDTLG
//
//    VARIABLES BELOW REPRESENT PROPERTIES EXISTING ONLY IN SATURATION
//    STATES, SUFFIX FG STANDS FOR SATURATION
//
global HFG SIGMAFG LACFG
//
//    VARIABLES BELOW REPRESENT MOST IMPORTANT QUANDITIES FOR THE MAIN
//    DATA-FITTING FUNCTION QMUST (PRIMARY CONSTANTS OF DOUBLE SERIES
//    EXPANSION OF Q
//
global Q QDT Q2D2T QTD Q2T2D Q2DT PSI0 PSI0T PSI02T2
```

II.2.34 Function viscosity (αρχείο viscosity.sci)

```
function[MU]=viscosity(TR,DD)
//
//    COMPUTE VISCOSITY OF HEAVY WATER IN kg/(ms) GIVEN TEMPERATURE
//    IN K UNITS AND DENSITY IN g/cm3
//    COMPUTATION AT VICINITY OF CRITICAL POINT IS NOT ACCURATE WITH
//    THIS FIT
//
exec('F:\scilab_hw\constants.sci');
exec('F:\scilab_hw\transient.sci');
A=[0.4864192D+0 0.3509007D+0 -0.2847572D+0 0.07013759D+0 0.01641220D+0 ...
-0.01163815D+0 0.D+0;-0.2448372D+0 1.315436D+0 ...
-1.037026D+0 0.4660127D+0 -0.02884911D+0 -0.008239587D+0 0.D+0;...
-0.8702035D+0 1.297752D+0 -1.287846D+0 0.2292075D+0 0.D+0 0.D+0 ...
0.D+0;0.8716056D+0 1.353448D+0 0.D+0 -0.4857462D+0 0.1607171D+0 0.D+0 ...
-0.003886659D+0;-1.051126D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0 0.D+0;...
0.3458395D+0 0.D+0 -0.02148229D+0 0.D+0 -0.009603846D+0 ...
0.004559914D+0 0.D+0];
B=[1.00000D+0 0.940695D+0 0.578377D+0 -0.202044D+0];
HETA=55.26516D-6;
PSEUDODR=DD/PSEUDODC;
NSIJ=0.;
NSK=0.;
for  I=1:6
      for  J=1:7
      NSIJ=NSIJ+A(I,J)*(((1./TR)-1)^(I-1))*((PSEUDODR-1)^(J-1));
      end
end
for K=1:4
      NSK=NSK+B(K)*((1./TR)^(K-1));
end
N0=HETA*(TR^0.5)*(1./NSK);
N=N0*exp(PSEUDODR*NSIJ);
MU=N;
return
endfunction
```

ΠΑΡΑΡΤΗΜΑ ΙΙΙ: Ενδεικτικά αποτελέσματα κωδίκων

ΙΙΙ.1 Κώδικας ελαφρού ύδατος - περιβάλλον MATLAB & SCILAB

Volume,m3/kg				Enthalpy,kJ/kg			Energy,kJ/kg			Entropy,kJ/kgK			
bar	4.0	4.2	4.4	4.0	4.2	4.4	4.0	4.2	4.4	4.0	4.2	4.4	bar
t(C)													t(C)
250	0.5951	0.5664	0.5402	2964.2	2963.5	2962.8	2726.1	2725.6	2725.1	7.3789	7.3554	7.3329	250
260	0.6071	0.5778	0.5512	2984.7	2984.1	2983.4	2741.9	2741.4	2740.9	7.4178	7.3944	7.3720	260
270	0.6191	0.5893	0.5621	3005.2	3004.6	3004.0	2757.6	2757.1	2756.7	7.4559	7.4326	7.4103	270
280	0.6311	0.6007	0.5730	3025.8	3025.2	3024.6	2773.3	2772.9	2772.5	7.4933	7.4701	7.4478	280
290	0.6430	0.6120	0.5839	3046.3	3045.7	3045.2	2789.1	2788.7	2788.3	7.5301	7.5069	7.4847	290
300	0.6548	0.6233	0.5947	3066.8	3066.3	3065.8	2804.8	2804.5	2804.1	7.5662	7.5430	7.5209	300
310	0.6667	0.6347	0.6055	3087.3	3086.8	3086.3	2820.6	2820.3	2819.9	7.6017	7.5786	7.5565	310
320	0.6785	0.6459	0.6163	3107.8	3107.4	3106.9	2836.4	2836.1	2835.8	7.6366	7.6135	7.5915	320
330	0.6903	0.6572	0.6271	3128.4	3128.0	3127.5	2852.3	2852.0	2851.6	7.6710	7.6480	7.6260	330
340	0.7021	0.6685	0.6378	3149.0	3148.6	3148.2	2868.2	2867.9	2867.5	7.7049	7.6819	7.6599	340
350	0.7139	0.6797	0.6486	3169.7	3169.3	3168.9	2884.1	2883.8	2883.5	7.7383	7.7153	7.6934	350
360	0.7257	0.6909	0.6593	3190.3	3190.0	3189.6	2900.1	2899.8	2899.5	7.7712	7.7482	7.7263	360
370	0.7374	0.7021	0.6700	3211.0	3210.7	3210.3	2916.1	2915.8	2915.5	7.8037	7.7807	7.7588	370
380	0.7492	0.7133	0.6807	3231.8	3231.5	3231.1	2932.1	2931.9	2931.6	7.8357	7.8128	7.7909	380

390	0.7609	0.7245	0.6913	3252.6	3252.3	3251.9	2948.2	2948.0	2947.8	7.8673	7.8444	7.8226	390
400	0.7726	0.7356	0.7020	3273.5	3273.1	3272.8	2964.4	2964.2	2963.9	7.8985	7.8756	7.8538	400
410	0.7843	0.7468	0.7127	3294.4	3294.1	3293.7	2980.6	2980.4	2980.2	7.9293	7.9065	7.8847	410
420	0.7960	0.7579	0.7233	3315.3	3315.0	3314.7	2996.9	2996.7	2996.5	7.9598	7.9369	7.9152	420
430	0.8077	0.7691	0.7340	3336.3	3336.0	3335.8	3013.2	3013.0	3012.8	7.9899	7.9670	7.9453	430
440	0.8194	0.7802	0.7446	3357.4	3357.1	3356.8	3029.6	3029.4	3029.2	8.0196	7.9968	7.9751	440
450	0.8311	0.7913	0.7552	3378.5	3378.2	3378.0	3046.1	3045.9	3045.7	8.0490	8.0262	8.0045	450
460	0.8427	0.8025	0.7658	3399.7	3399.4	3399.2	3062.6	3062.4	3062.2	8.0781	8.0553	8.0336	460
470	0.8544	0.8136	0.7764	3420.9	3420.7	3420.4	3079.1	3079.0	3078.8	8.1068	8.0841	8.0624	470
480	0.8661	0.8247	0.7871	3442.2	3441.9	3441.7	3095.8	3095.6	3095.4	8.1353	8.1125	8.0909	480
490	0.8777	0.8358	0.7977	3463.5	3463.3	3463.1	3112.4	3112.3	3112.1	8.1635	8.1407	8.1190	490

ΙΙΙ.2 Απόσπασμα ιδιοτήτων από πίνακες ελαφρού ύδατος

	4,0 bar t_s = 143,62 °C			4,2 bar t_s = 145,39 °C			4,4 bar t_s = 147,09 °C			4,6 bar t_s = 148,73 °C		
z	v'' 0,4622	h'' 2737,6	s'' 6,8943	v'' 0,4415	h'' 2739,8	s'' 6,8779	v'' 0,4276	h'' 2741,9	s'' 6,8613	v'' 0,4053	h'' 2743,9	s'' 6,8473
°C	v	h	s	v	h	s	v	h	s	v	h	s
0	0,0010000	0,4	−0,0001	0,0010000	0,4	−0,0001	0,0010000	0,4	−0,0001	0,0010000	0,4	−0,0001
10	0,0010001	42,4	0,1510	0,0010001	42,4	0,1510	0,0010000	42,4	0,1510	0,0010000	42,4	0,1510
20	0,0010015	84,1	0,2962	0,0010015	84,3	0,2962	0,0010015	84,3	0,2962	0,0010015	84,3	0,2962
30	0,0010041	126,0	0,4364	0,0010041	126,0	0,4364	0,0010041	126,1	0,4364	0,0010043	126,1	0,4364
40	0,0010076	167,8	0,5720	0,0010076	167,8	0,5710	0,0010076	167,8	0,5710	0,0010076	167,9	0,5719
50	0,0010119	209,6	0,7033	0,0010119	209,6	0,7033	0,0010119	209,6	0,7033	0,0010119	209,6	0,7033
60	0,0010170	251,4	0,8308	0,0010170	251,4	0,8308	0,0010170	251,4	0,8308	0,0010169	251,5	0,8307
70	0,0010227	293,3	0,9546	0,0010227	293,3	0,9546	0,0010227	293,3	0,9546	0,0010226	293,3	0,9546
80	0,0010290	335,2	1,0750	0,0010290	335,2	1,0750	0,0010290	335,2	1,0750	0,0010290	335,2	1,0750
90	0,0010360	377,2	1,1923	0,0010360	377,2	1,1923	0,0010360	377,2	1,1923	0,0010360	377,2	1,1922
100	0,0010436	419,3	1,3066	0,0010435	419,3	1,3066	0,0010435	419,3	1,3066	0,0010433	419,3	1,3066
110	0,0010517	461,5	1,4183	0,0010517	461,5	1,4183	0,0010517	461,5	1,4182	0,0010517	461,5	1,4181
120	0,0010605	503,9	1,5274	0,0010605	503,9	1,5274	0,0010605	503,9	1,5274	0,0010605	503,9	1,5274
130	0,0010699	546,4	1,6342	0,0010699	546,4	1,6342	0,0010699	546,4	1,6342	0,0010699	546,4	1,6342
140	0,0010800	589,1	1,7389	0,0010800	589,1	1,7389	0,0010800	589,2	1,7389	0,0010800	589,2	1,7389
150	0,4707	2752,0	6,9285	0,4473	2750,3	6,9028	0,4261	2748,5	6,8780	0,4068	2746,8	6,8542
160	0,4837	2774,2	6,9805	0,4599	2772,7	6,9551	0,4382	2771,1	6,9308	0,4184	2769,5	6,9074
170	0,4966	2796,1	7,0305	0,4722	2794,7	7,0055	0,4500	2793,3	6,9815	0,4297	2791,9	6,9585
180	0,5093	2817,8	7,0788	0,4843	2816,5	7,0541	0,4617	2815,2	7,0304	0,4410	2814,0	7,0077
190	0,5218	2839,2	7,1255	0,4963	2838,0	7,1010	0,4732	2836,9	7,0776	0,4520	2835,7	7,0552
200	0,5343	2860,4	7,1708	0,5082	2859,3	7,1466	0,4846	2858,3	7,1234	0,4630	2857,2	7,1011
210	0,5466	2881,4	7,2148	0,5200	2880,5	7,1908	0,4959	2879,5	7,1678	0,4738	2878,6	7,1457
220	0,5589	2902,3	7,2576	0,5317	2901,5	7,2338	0,5071	2900,6	7,2109	0,4846	2899,7	7,1891
230	0,5710	2923,1	7,2994	0,5434	2922,3	7,2757	0,5182	2921,5	7,2530	0,4953	2920,7	7,2312
240	0,5831	2943,9	7,3402	0,5549	2943,2	7,3166	0,5293	2942,4	7,2940	0,5059	2941,6	7,2724
250	0,5952	2964,5	7,3800	0,5664	2963,8	7,3565	0,5403	2963,1	7,3340	0,5164	2962,4	7,3125
260	0,6072	2985,1	7,4190	0,5779	2984,5	7,3956	0,5513	2983,8	7,3732	0,5269	2983,2	7,3518
270	0,6192	3005,6	7,4572	0,5893	3005,1	7,4339	0,5622	3004,5	7,4116	0,5374	3003,9	7,3902
280	0,6311	3026,2	7,4947	0,6007	3025,6	7,4714	0,5731	3025,1	7,4492	0,5478	3024,5	7,4279
290	0,6430	3046,7	7,5314	0,6120	3046,2	7,5082	0,5839	3045,7	7,4861	0,5582	3045,2	7,4649
300	0,6549	3067,2	7,5675	0,6234	3066,7	7,5444	0,5947	3066,2	7,5223	0,5686	3065,7	7,5011
310	0,6667	3087,7	7,6030	0,6346	3087,3	7,5799	0,6055	3086,8	7,5579	0,5789	3086,3	7,5368
320	0,6785	3108,3	7,6379	0,6459	3107,8	7,6149	0,6163	3107,4	7,5929	0,5892	3107,0	7,5718
330	0,6903	3128,8	7,6723	0,6571	3128,4	7,6493	0,6270	3128,0	7,6273	0,5995	3127,6	7,6063
340	0,7021	3149,4	7,7061	0,6684	3149,0	7,6831	0,6378	3148,6	7,6612	0,6098	3148,2	7,6402
350	0,7139	3170,0	7,7395	0,6796	3169,6	7,7165	0,6485	3169,2	7,6946	0,6201	3168,9	7,6736
360	0,7256	3190,6	7,7723	0,6908	3190,3	7,7494	0,6592	3189,9	7,7275	0,6303	3189,6	7,7066
370	0,7373	3211,3	7,8047	0,7020	3211,0	7,7818	0,6699	3210,6	7,7600	0,6406	3210,3	7,7391
380	0,7491	3232,1	7,8367	0,7131	3231,7	7,8138	0,6806	3231,4	7,7920	0,6508	3231,1	7,7711
390	0,7608	3252,8	7,8683	0,7244	3252,5	7,8454	0,6912	3252,2	7,8236	0,6610	3251,9	7,8027
400	0,7725	3273,6	7,8994	0,7355	3273,3	7,8766	0,7019	3273,0	7,8548	0,6712	3274,7	7,8339
410	0,7842	3294,5	7,9302	0,7467	3294,2	7,9074	0,7125	3293,9	7,8856	0,6814	3293,6	7,8648
420	0,7959	3315,4	7,9606	0,7578	3315,1	7,9378	0,7232	3314,9	7,9160	0,6916	3314,6	7,8952
430	0,8076	3336,4	7,9906	0,7689	3336,1	7,9678	0,7338	3333,9	7,9461	0,7018	3335,6	7,9253
440	0,8192	3357,4	8,0203	0,7801	3357,2	7,9976	0,7444	3356,9	7,9738	0,7119	3356,6	7,9550
450	0,8309	3378,5	8,0497	0,7911	3378,3	8,0269	0,7551	3378,0	8,0051	0,7221	3377,7	7,9844
460	0,8426	3399,7	8,0787	0,8023	3399,4	8,0560	0,7657	3399,2	8,0343	0,7322	3398,9	8,0135
470	0,8542	3420,9	8,1075	0,8134	3420,6	8,0847	0,7763	3420,4	8,0630	0,7424	3420,2	8,0423
480	0,8659	3442,1	8,1359	0,8245	3441,9	8,1132	0,7869	3441,7	8,0915	0,7526	3441,4	8,0707
490	0,8775	3463,5	8,1640	0,8356	3463,2	8,1413	0,7975	3463,0	8,1196	0,7627	3462,8	8,0989
500	0,8892	3484,9	8,1919	0,8467	3484,6	8,1692	0,8081	3484,4	8,1475	0,7728	3484,2	8,1268
510	0,9008	3506,3	8,2195	0,8578	3506,1	8,1967	0,8187	3505,9	8,1751	0,7830	3505,7	8,1544
520	0,9125	3527,8	8,2468	0,8689	3527,6	8,2241	0,8293	3527,4	8,2024	0,7931	3527,2	8,1817
530	0,9241	3549,4	8,2738	0,8800	3549,2	8,2511	0,8399	3549,0	8,2294	0,8032	3548,8	8,2087
540	0,9357	3571,2	8,3006	0,8911	3570,9	8,2779	0,8504	3570,7	8,2563	0,8134	3570,5	8,2356
550	0,9474	3592,8	8,3271	0,9021	3592,6	8,3044	0,8610	3592,4	8,2828	0,8235	3592,2	8,2611

III.3 Κώδικας βαρέος ύδατος - περιβάλλον MATLAB & SCILAB

	Volume,m3/kg			Enthalpy,kJ/kg			Energy,kJ/kg			Entropy,kJ/kgK			
bar	4.5	5.0	5.5	4.5	5.0	5.5	4.5	5.0	5.5	4.5	5.0	5.5	bar
t(C)													t(C)
275	0.4990	0.4484	0.4070	2797.8	2796.4	2795.0	2573.3	2572.2	2571.2	6.8816	6.8360	6.7945	275
280	0.5038	0.4527	0.4110	2807.7	2806.3	2805.0	2581.0	2580.0	2579.0	6.8995	6.8540	6.8126	280
285	0.5086	0.4570	0.4149	2817.5	2816.2	2814.9	2588.7	2587.7	2586.8	6.9173	6.8718	6.8305	285
290	0.5133	0.4614	0.4189	2827.4	2826.2	2824.9	2596.4	2595.5	2594.5	6.9349	6.8895	6.8483	290
295	0.5181	0.4657	0.4228	2837.3	2836.1	2834.9	2604.1	2603.2	2602.3	6.9524	6.9070	6.8659	295
300	0.5229	0.4700	0.4267	2847.2	2846.0	2844.8	2611.9	2611.0	2610.1	6.9697	6.9244	6.8833	300
305	0.5276	0.4743	0.4306	2857.1	2855.9	2854.8	2619.6	2618.8	2617.9	6.9869	6.9417	6.9006	305
310	0.5324	0.4786	0.4346	2867.0	2865.9	2864.7	2627.4	2626.6	2625.7	7.0039	6.9588	6.9178	310
315	0.5372	0.4829	0.4385	2876.9	2875.8	2874.7	2635.2	2634.4	2633.6	7.0208	6.9757	6.9348	315
320	0.5419	0.4872	0.4424	2886.8	2885.7	2884.7	2642.9	2642.2	2641.4	7.0376	6.9926	6.9517	320
325	0.5466	0.4914	0.4463	2896.7	2895.7	2894.7	2650.7	2650.0	2649.2	7.0543	7.0093	6.9685	325
330	0.5514	0.4957	0.4502	2906.7	2905.7	2904.7	2658.5	2657.8	2657.1	7.0709	7.0259	6.9851	330
335	0.5561	0.5000	0.4541	2916.6	2915.6	2914.7	2666.4	2665.6	2664.9	7.0873	7.0424	7.0016	335
340	0.5608	0.5043	0.4580	2926.6	2925.6	2924.7	2674.2	2673.5	2672.8	7.1036	7.0587	7.0180	340
345	0.5656	0.5085	0.4619	2936.5	2935.6	2934.7	2682.0	2681.4	2680.7	7.1198	7.0750	7.0343	345

350	0.5703	0.5128	0.4657	2946.5	2945.6	2944.8	2689.9	2689.3	2688.6	7.1359	7.0911	7.0505	350
355	0.5750	0.5170	0.4696	2956.5	2955.7	2954.8	2697.8	2697.1	2696.5	7.1519	7.1071	7.0665	355
360	0.5797	0.5213	0.4735	2966.5	2965.7	2964.9	2705.7	2705.1	2704.4	7.1678	7.1230	7.0825	360
365	0.5844	0.5255	0.4774	2976.6	2975.8	2974.9	2713.6	2713.0	2712.4	7.1835	7.1388	7.0983	365
370	0.5891	0.5298	0.4812	2986.6	2985.8	2985.0	2721.5	2720.9	2720.3	7.1992	7.1546	7.1141	370
375	0.5938	0.5340	0.4851	2996.7	2995.9	2995.1	2729.5	2728.9	2728.3	7.2148	7.1702	7.1297	375
380	0.5985	0.5383	0.4890	3006.8	3006.0	3005.2	2737.4	2736.9	2736.3	7.2303	7.1857	7.1452	380
385	0.6032	0.5425	0.4928	3016.8	3016.1	3015.3	2745.4	2744.8	2744.3	7.2457	7.2011	7.1607	385
390	0.6079	0.5467	0.4967	3026.9	3026.2	3025.5	2753.4	2752.8	2752.3	7.2610	7.2164	7.1760	390
395	0.6126	0.5510	0.5005	3037.1	3036.4	3035.6	2761.4	2760.9	2760.3	7.2762	7.2316	7.1913	395

ΙΙΙ.4 Απόσπασμα ιδιοτήτων από πίνακες βαρέος ύδατος

	Volume, m^3/kg			Enthalpy, kJ/kg			Energy, kJ/kg			Entropy, kJ/kg.K			
p(kPa)	450.	500	550	450.	500	550	450	500	550	450	500	550.	p(kPa)
T(C)													T(C)
275	0.4990	0.4484	0.4070	2797.9	2796.4	2795.0	2673.3	2672.2	2671.2	6.8816	6.8360	6.7945	275
280	0.5036	0.4527	0.4110	2807.7	2806.3	2805.0	2683.0	2680.0	2679.0	6.8935	6.8540	6.8126	280.
285	0.5086	0.4570	0.4149	2817.5	2816.2	2814.9	2688.7	2687.7	2686.8	6.9173	6.8718	6.8305	285
290.	0.5133	0.4614	0.4189	2827.4	2826.2	2824.9	2696.4	2695.5	2694.5	6.9349	6.8895	6.8483	290
295	0.5181	0.4657	0.4228	2837.3	2836.1	2834.9	2604.1	2603.2	2602.3	6.9534	6.9070	6.8659	295.
300	0.5229	0.4700	0.4267	2847.2	2846.0	2844.8	2611.9	2611.0	2610.1	6.9637	6.9244	6.8830	300
305	0.5276	0.4743	0.4306	2857.1	2855.9	2854.8	2619.6	2618.9	2617.9	6.9869	6.9417	6.9006	305
310	0.5324	0.4786	0.4346	2867.0	2865.9	2864.7	2627.4	2626.6	2625.7	7.0039	6.9588	6.9176	310
315	0.5371	0.4829	0.4385	2876.9	2875.9	2874.7	2635.2	2634.4	2633.6	7.0208	6.9757	6.9346	315
320	0.5419	0.4872	0.4424	2886.8	2885.7	2884.7	2642.9	2642.2	2641.4	7.0376	6.9926	6.9517	320
325	0.5466	0.4914	0.4463	2896.7	2895.7	2894.7	2650.7	2650.0	2649.2	7.0543	7.0093	6.9685	325
330	0.5514	0.4957	0.4502	2906.7	2905.7	2904.7	2658.5	2657.8	2657.1	7.0708	7.0259	6.9851	330
335	0.5561	0.5000	0.4541	2916.6	2915.6	2914.7	2666.3	2665.6	2664.9	7.0873	7.0424	7.0016	335
340.	0.5608	0.5043	0.4580	2926.6	2925.6	2924.7	2674.2	2673.5	2672.8	7.1036	7.0587	7.0180	340.
345.	0.5656	0.5085	0.4619	2936.5	2935.6	2934.7	2682.0	2681.4	2680.7	7.1198	7.0750	7.0343	345.
350.	0.5703	0.5128	0.4657	2946.5	2945.6	2944.8	2689.9	2689.3	2688.6	7.1359	7.0911	7.0505	350.
355	0.5750	0.5170	0.4696	2956.5	2955.7	2954.8	2697.9	2697.3	2696.5	7.1519	7.1071	7.0665	355.
360	0.5797	0.5213	0.4735	2966.5	2965.7	2964.9	2705.7	2705.1	2704.4	7.1678	7.1230	7.0825	360
365	0.5844	0.5255	0.4774	2976.6	2975.8	2974.9	2713.6	2713.0	2712.3	7.1835	7.1388	7.0982	365
370	0.5891	0.5298	0.4812	2986.6	2985.8	2985.0	2721.5	2720.9	2720.3	7.1992	7.1545	7.1141	370
375	0.5938	0.5340	0.4851	2996.7	2995.9	2995.1	2729.5	2728.9	2728.3	7.2148	7.1702	7.1297	375
380	0.5985	0.5383	0.4890	3006.8	3006.0	3005.2	2737.4	2736.9	2736.3	7.2303	7.1857	7.1452	380
385	0.6032	0.5425	0.4928	3016.9	3016.1	3015.3	2745.4	2744.8	2744.3	7.2457	7.2011	7.1607	385
390	0.6079	0.5467	0.4967	3026.9	3026.2	3025.5	2753.4	2752.8	2752.3	7.2610	7.2164	7.1760	390
395	0.6126	0.5510	0.5005	3037.1	3036.3	3035.6	2761.4	2760.9	2760.3	7.2762	7.2316	7.1913	395
400	0.6173	0.5552	0.5044	3047.2	3046.5	3045.8	2769.4	2768.9	2768.4	7.2913	7.2468	7.2064	400
410	0.6267	0.5636	0.5121	3067.5	3066.8	3066.2	2785.5	2785.0	2784.5	7.3212	7.2768	7.2365	410
420	0.6360	0.5721	0.5198	3087.9	3087.3	3086.6	2801.7	2801.2	2800.8	7.3509	7.3064	7.2662	420.
430	0.6454	0.5805	0.5274	3108.4	3107.8	3107.1	2817.9	2817.5	2817.0	7.3802	7.3358	7.2956	430
440	0.6547	0.5890	0.5351	3128.9	3128.3	3127.7	2834.3	2833.8	2833.4	7.4092	7.3648	7.3246	440.
450	0.6641	0.5974	0.5428	3149.5	3148.9	3148.4	2850.7	2850.2	2849.8	7.4379	7.3935	7.3534	450.
460	0.6734	0.6058	0.5504	3170.2	3169.6	3169.1	2867.3	2866.7	2866.3	7.4662	7.4220	7.3818	460
470	0.6828	0.6142	0.5581	3190.9	3190.4	3189.9	2883.9	2883.5	2883.1	7.4943	7.4501	7.4100	470
480	0.6921	0.6226	0.5658	3211.7	3211.2	3210.7	2900.7	2899.9	2899.6	7.5222	7.4779	7.4379	480
490	0.7014	0.6310	0.5734	3232.6	3232.1	3231.7	2917.0	2916.5	2916.3	7.5497	7.5055	7.4655	490
500	0.7107	0.6394	0.5810	3253.6	3253.1	3252.7	2933.8	2933.4	2933.1	7.5770	7.5329	7.4929	500
510	0.7200	0.6478	0.5887	3274.7	3274.2	3273.7	2950.6	2950.3	2950.0	7.6041	7.5599	7.5199	510
520	0.7294	0.6562	0.5963	3295.8	3295.3	3294.9	2967.5	2967.2	2966.9	7.6309	7.5867	7.5468	520
530.	0.7387	0.6646	0.6039	3317.0	3316.6	3316.1	2984.6	2984.3	2983.9	7.6575	7.6133	7.5734	530.
540	0.7480	0.6730	0.6116	3338.2	3337.8	3337.4	3001.7	3001.4	3001.1	7.6838	7.6397	7.5997	540
550	0.7573	0.6813	0.6192	3359.6	3359.2	3358.9	3018.9	3018.5	3018.2	7.7099	7.6658	7.6258	550
560	0.7666	0.6897	0.6268	3381.1	3380.7	3380.3	3036.3	3035.9	3035.6	7.7359	7.6917	7.6518	560
570	0.7759	0.6981	0.6344	3402.6	3402.2	3401.8	3053.7	3053.4	3053.1	7.7616	7.7174	7.6774	570
580	0.7852	0.7064	0.6421	3424.2	3423.8	3423.4	3071.3	3070.9	3070.6	7.7869	7.7428	7.7029	580
590	0.7944	0.7148	0.6497	3445.8	3445.5	3445.1	3088.9	3088.6	3088.3	7.8121	7.7681	7.7282	590
600	0.8037	0.7232	0.6573	3467.6	3467.2	3466.8	3106.6	3106.3	3106.0	7.8372	7.7931	7.7532	600
620	0.8223	0.7399	0.6725	3511.3	3510.9	3510.6	3141.9	3141.6	3141.3	7.8867	7.8427	7.8028	620
640	0.8409	0.7566	0.6877	3555.2	3554.9	3554.7	3177.4	3177.1	3176.8	7.9354	7.8914	7.8516	640
660	0.8594	0.7733	0.7029	3599.4	3599.1	3598.9	3213.1	3212.8	3212.6	7.9835	7.9395	7.8996	660
680	0.8780	0.7900	0.7181	3644.0	3643.8	3643.7	3249.1	3248.9	3248.7	8.0308	7.9868	7.9470	680
700	0.8965	0.8067	0.7332	3689.1	3688.9	3688.8	3285.3	3285.1	3285.0	8.0774	8.0335	7.9937	700
720	0.9151	0.8234	0.7484	3734.6	3734.5	3734.4	3321.8	3321.6	3321.5	8.1234	8.0795	8.0397	720
740	0.9336	0.8401	0.7637	3780.7	3780.6	3780.5	3358.6	3358.5	3358.4	8.1688	8.1248	8.0851	740
760	0.9521	0.8568	0.7788	3827.3	3827.2	3827.2	3395.7	3395.6	3395.5	8.2136	8.1696	8.1299	760
780	0.9706	0.8735	0.7940	3874.5	3874.5	3874.5	3433.0	3433.0	3432.9	8.2579	8.2138	8.1740	780
800	0.9892	0.8902	0.8092	3922.3	3922.3	3922.3	3470.7	3470.6	3470.6	8.3016	8.2574	8.2177	800